CORE-CRUST MODELING APPROACH FOR FORMAL REPRESENTATION OF TRUST IN RELATION TO COMPUTER SECURITY

by

Şerif Bahtiyar

B.S., Control and Computer Engineering, Istanbul Technical University, 2001

M.S., Computer Engineering, Istanbul Technical University, 2004

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

Graduate Program in Computer Engineering

Boğaziçi University

2011

ACKNOWLEDGEMENTS

There are many people I would like to thank due to their support, encouragement, and trust during my doctoral study. In particular, I would like to express my deepest appreciation and sincere to my supervisor Prof. Mehmet Ufuk Çağlayan. His guidance, understanding, criticism and support were inspiring and motivational through the good and the bad times. I also thank to him for organizing the financial support required for me to pursue my doctoral study at Boğaziçi University. This thesis would not be possible without him. My thanks also go to my thesis jury members Assoc. Prof. Fatih Alagöz, Assist. Prof. Alper Şen, Assoc. Prof. Pınar Yolum Birbil, Prof. Çetin Kaya Koç, and Assoc. Prof. Albert Levi for spending their rare available time on examining my thesis.

I would also like to thank my colleagues and friends at the Computer Networks Research Laboratory who made my time with them enjoyable. In particular, I would like to thank Murat Cihan, Devrim Ünal, Orhan Ermiş, A. Burak Gürdağ, Can Komar, E. Itır Karaç, and A. Haydar Özer for several insightful discussions and suggestions. My apologies go to those I may miss, a collective thanks to all.

I am grateful to the Turkish State Planning Organization (DPT) under the TAM Project, number 2007K120610 for supporting my research on modeling and formal representation of trust in relation to computer security.

Last but not least I would like to thank my family. My parents, Halime and Mehmet Emin have always encouraged and supported me when I took important decisions and always been with me when I needed them. Thanks to my sister Şenay for her moral support. My thanks also go to my grandfather Şerif. Finally, my deepest thanks go to my wife, Gülhan and my son Mete. Their love, support, care, and very existence are my keystone. Without their patience and encouragement, this thesis would never have been finished.

ABSTRACT

CORE-CRUST MODELING APPROACH FOR FORMAL REPRESENTATION OF TRUST IN RELATION TO COMPUTER SECURITY

The increasing availability of high bandwidth network connections with low-cost computing equipments has stimulated the use of service-oriented environments. Emerging service-oriented environments are expected to be open environments. Such environments are highly dynamic and contain diverse number of services and autonomous entities. However, their openness reveals trust problems related to security systems. To cope with the problems, precise models and representations of trust and security are needed. This thesis examines, models, and represents trust in relation to computer security in emerging open environments with core-crust modeling approach. The thesis contains four main contributions. First, we introduce the core model for trust assessment of the security system of a service from an entity point of view. The model could be applied to almost all entities in open environments. As the second main contribution, we propose a crust model for extracting trust information from the security system of a service, based on needs of a specific entity. Our next main contribution is a crust model for trust assessment based on flow of security evaluation information on entities. The aim of this model is to increase the amount of security evaluation information to be used in trust assessments. Finally, we propose a crust model for trust based security interoperability for service convergence in networks. The last model aims to show how our proposed models can be integrated to our core model for trust assessments. We also show the applicability of each model with case studies and simulations.

ÖZET

BİLGİSAYAR GÜVENLİĞİNDE GÜVENİN ÇEKİRDEK KABUK MODELLEME YAKLAŞIMI İLE KURALLI GÖSTERİMİ

Düşük maliyetli bilgisayar ekipmanları ile yüksek bant genişliğine sahip ağ bağlantılarının giderek artması, hizmet odaklı ortamların da kullanımının artmasına neden olmaktadır. Gelişen hizmet odaklı ortamların açık ortamlar olması beklenmektedir. Bu tür ortamlar son derece dinamiktirler ve çeşitli servisler ile otonom etmenler içerirler. Ancak, ortamların açıklığı, güvenlik sistemleri ile ilgili güven problemlerine neden olmaktadır. Bu problemlerle başa çıkmak için güven ve güvenliğin hassas olarak modellenmesine ve kurallı olarak gösterimine ihtiyaç vardır. Bu tez, çekirdek kabuk yaklaşımı ile açık ortamlardaki bilgisayar güvenliği ile ilgili güveni inceler, modeller ve nasıl temsil edileceğini gösterir. Bu tezin dört ana katkısı vardır. Öncelikle, bir etmenin bakış açısından bir servisin güvenlik sisteminin güvenini değer biçecek çekirdek modeli sunuyoruz. Bu model açık ortamlardaki neredeyse tüm etmenlere uygulanabilir. İkinci olarak, belirli bir etmenin ihtiyaçlarına göre bir servisin güvenlik sisteminden güven bilgisini çıkarmak için bir kabuk model öneriyoruz. Sonraki ana katkımız, etmenler üzerinden geçen güvenlik değerlendirme bilgisine göre güveni tayin eden bir kabuk modeldir. Bu modelin amacı, güven değerlendirmelerinde kullanılmak üzere güvenlik değerlendirme bilgi miktarını arttırmaktır. Son olarak, ağlarda servis yakınsaması için güvene dayalı güvenliğin birlikte çalışabilirliği sağlayacak bir kabuk model öneriyoruz. Son model, güven değerlendirmeleri için modellerin bizim temel modelimize nasıl ekleneceğini gösteriyor. Ayrıca, her modelin uygulanabilirliğini örnek olaylar ve benzetimler ile gösteriyoruz.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

c^f	The coefficient of false experienced atomic units
c^r	The coefficient of true experienced atomic units
co_x^α	The coefficient of $\iota_x^\alpha(t)$
$co_{x,y}^\alpha$	The coefficient of $\iota_{x,y}^\alpha(t)$
co_x^ω	The coefficient of $\iota_x^\omega(t)$
$co_{x,y}^\omega$	The coefficient of $\iota_{x,y}^\omega(t)$
co_x^χ	The coefficient of $\iota_x^\chi(t)$
$co_{x,y}^\chi$	The coefficient of $\iota_{x,y}^\chi(t)$
$efSim$	The effective similarity ratio
$f_\alpha^{sim}()$	Similarity function of entity α
h_j	The overall history effect on atomic unit $p_j \in P_c^{xs}$
h_j^{sss}	The combined histories of atomic units in Φ^x related to atomic unit p_j
$h_{j,i}^{sss}$	The effect of history of atomic unit φ_i in relation to atomic unit p_j
imp_k	The impact factor of atomic unit $p_k \in P_c^e$
$iDif$	Information Difference
k	Importance Factor
k_s	The quality of ultimate experience ϵ_s
k_s^ω	Service Awareness
n_e	The number of entities interacting with a specific entity
n_s	The number services interacting with a specific entity
n_{fc}	Maximum number of former trust assessments
p	An atomic unit of a security policy representation
r	An interaction
r_s	Service Reputation
stp_i	The satisfaction factor of atomic unit p_i of a security policy
sts_i	The satisfaction factor of atomic unit φ_i of a security system
t_i^a	The assessed trust related to the security system of service ω_i

ta_j	Trust information related to atomic unit $p_j \in P_c^{xs}$
tr_s^a	The assessed trust about the security system of service ω_s
tr_s^m	The minimum trust about the security system of service ω_s
$u_\alpha\,()$	The utility function of entity α
v_j	Existence of experience about atomic unit φ_j
A	Set of entities
A_r	Set of entities from which recommendations are received
A_s	Set of entities to which recommendations are sent
N	The most recent n instances of ultimate experiences and ultimate recommendations that contribute to the computation of the confidence of a recommendation
$O\,()$	Computational Complexity
P_c^e	Set representation of the security policy of an entity related to service ω_c
P_c^s	Set representation of the security policy of service ω_c
P_c^{xs}	Set representation of the expected security policy of service ω_c in an entity
R	Set of interactions
V_s	Experience existence set of service ω_s.
W	Maximum number of services or required number of services
α	Entity
γ	Aging Factor of recommendations
$\gamma_{x,y}^{PC}$	Partial Confidence of $\gamma_{x,y}^{PT}$
$\gamma_{x,y}^{PR}$	Partial Relative Trust Assessment of atomic unit $\varphi_y \in \Phi_x$
$\gamma_{x,y}^{PT}$	Partial Trust Level of atomic unit $\varphi_y \in \Phi_x$
γ_x^{TC}	Total Confidence of γ_x^{TT}
γ_x^{TR}	Total Relative Trust Assessment about the security system of service ω_x
γ_x^{TT}	Total Trust Level about the security system of service ω_x
δ_i	Sent confidence about the recommendation related to the security system of service ω_i

$\delta^r_{d,i}$	Received confidence about the recommendation received from entity α_d related to the security system of service ω_i
ϵ_i	Ultimate experience related to service ω_i
ϵ^a_i	Aggregated experience related to service ω_i
ϵ^r_i	Obtained experience from service ω_i
$\zeta_{x,y}$	Impact Factor about atomic unit $\varphi_y \in \Phi_x$
η	Acceptance Threshold
θ^r	Risk Factor
θ^s	Accuracy Factor
ι	Extracted trust information related to P^e
ι_k	Extracted trust information related to atomic unit $p_k \in P^e_c$
ι^α_x	Total perceived information about the security system of service ω_x
$\iota^\alpha_{x,y}$	Partial perceived information about atomic unit $\varphi_y \in \Phi_x$
ι^ω_x	Total service information about the security system of service ω_x
$\iota^\omega_{x,y}$	Partial service information about atomic unit $\varphi_y \in \Phi_x$
ι^χ_x	Total information of former assessments about the security system of service ω_x
$\iota^\chi_{x,y}$	Partial information of former assessments about atomic unit $\varphi_y \in \Phi_x$
$\kappa_{x,y}$	The significance of recent information
λ	Similarity Ratio
μ_i	Ultimate recommendation related to service ω_i
μ^a_i	Aggregated recommendation related to service ω_i
μ^T_i	Total recommendation related to service ω_i
$\mu^p_{d,i}$	Perceived recommendation related to service ω_i received from entity α_d
$\mu^r_{d,i}$	Received recommendation from entity α_d about the security system of service ω_i
$\mu^s_{d,i}$	Sent recommendation to entity α_d about the security system of service ω_i

$\pi_{k,j}$	The perception factor of atomic unit $p_j \in P_c^{xs}$ related to atomic unit $p_k \in P_c^e$
σ_i	Reliability Factor about the security system of service ω_i
τ	Aging Factor of experiences
φ	An atomic unit
ω	Service
Λ_i	Set of similarity ratios related to service ω_i
Φ	Set of atomic units in an entity
Φ_c^x	Set representation of expected security system of service ω_c
Φ_c^r	The set of true experienced atomic units about the security system of service ω_c
Φ_c^f	The set of false experienced atomic units about the security system of service ω_c
$\Psi_{x,y}$	The effect of former assessments about atomic unit $\varphi_y \in \Phi_x$
Ω	Set of services
Ω^{max}	Maximum interacted set

LIST OF ACRONYMS/ABBREVIATIONS

AC	Access Control Mechanism
AES128	Advanced Encryption Standard with key size of 128 bits
AES256	Advanced Encryption Standard with key size of 256 bits
BU	Bogazici University
CCMA	Core Crust Modeling Approach
DES	Data Encryption Standard
DSS	Digital Signature Standard
ENC	Encryption algorithms, DES, AES128, and AES256
PA	Software Agent of a patient
PW	Password based authentication
SA	Software Agent
TAS	Trust Assessment System
TPM	Trusted Platform Module

1. INTRODUCTION

Nourishment, security, and procreation are some of the essential requirements for every living on the world. All livings interact with each other to cope with their basic needs of life. Interactions between human beings may be one of the most complex one among all other living beings. Generally, the interaction between two people or any two animals depends on their senses and feelings.

Humans use some artifacts to interact with each other. Computers, networks, software systems are some of the artifacts. They improve our quality of life by realizing some tasks and expressing our senses and feelings to other people. Human senses mostly depend on physical quantities such as heat, voice, light. These quantities can be easily represented by artifacts. However, feelings, such as fear, trust, risk, reputation, do not completely depend on physical quantities. Feelings are subjective quantities that may differ from one person to another one. Contemporary artifacts represent many senses successfully, but they are not able to represent and deal with the feelings well.

On the other hand, people create and publish too much information related to any topic that complicates and moreover makes impossible for an ordinary person to make decision about a specific topic. Too much information is known as "information overload" [1] that refers to the state of having too much information to make a decision. Actually, it may seem good to have vast choices about a topic to make a decision, but in reality, it is impossible to examine all alternatives. Furthermore, information is generally published by people who do not have direct interaction with a person who intends to use the created information. Therefore, reliability problems occur. If the person does not have any opinion about reliability of the intended information, it is not meaningful to use this information. Therefore, we need a way to cope with this problem.

The relations among people are organized by laws, rules and policies. A law is a collection of rules that is enforced by governments or any institution and that

must compel or prohibit behaviors. On the other hand, a policy is a set of high level rules that are not strictly enforced [2]. In real life, we have lots of policies, such as economic policies, defense policies, education policies, security policies, communication policies. A security policy is described as a collection of clearly defined rules that allow or disallow possible actions, events, or something related to security. In this thesis, we are concerned with security policies related to computer science. Various definitions of security policy are given in [2–8].

The role of security policies in contemporary computing platforms and information systems has been increasing day by day. Therefore, there are many types of security policies and security policy development strategies [9]. Contemporary security policies are dynamic and they may be modified quite frequently. Therefore, the complexity of security policies increases, which results in new security problems.

Enforcement of security policies introduces additional security problems. The development of security mechanisms is a complex task and it necessitates expertise of many people. Moreover, security mechanisms may frequently be modified based on new demands. Therefore, the complexity and the dynamic nature of security mechanisms may reveal some additional security problems, such as policy enforcement in clients in distributed systems [10].

Emerging open environments are highly dynamic and they contain diverse number of services and autonomous entities. Entities in such environments have different security needs from services. Most of the time, an entity interacts with a service if the entity trusts to the security system of the service. Otherwise, the entity may prefer to interact with other services or it may simply give up interacting with the service.

Assessing the trust to security systems of services based on needs of an entity is a significant task for trust based decision making. Trust computations related to the security systems of services necessitate information that meets needs of each entity. Obtaining such information is a challenging issue for entities. On the other hand, the trust computations necessitate a formal representation of information on entities

because of subjective nature of trust, such as modeling critical systems [11–16]. Moreover, trust models related to information gathering and trust assessment about security systems of services based on private needs of entities are needed for making trust based decisions.

1.1. Motivation

There are many trust computation models that can be applied for assessing the trust of the security system of a service [5,17–19]. However, modeling and representing trust of the security system of a service based on needs of each entity is not clearly addressed in these models. Specifically, trust has a subjective nature. Therefore, a trust model may be convenient for an entity or a service in an open environment, but the model may be inconvenient for another entity or another service in the environment.

Entities and services should interact with each other to accomplish their tasks based on trust decision making. For this reason, such entities should have common trust models to be able to make trust decisions. Particularly, uncertainty is not good for security based decision making [20]. Trust decisions that are based on security systems of services should be made according to a certain trust model. Therefore, entities necessitate precise representation of trust for making trust decisions related to the security system of a service in open environments. These models should not be changed frequently so entities can have certain and common trust models, which is good for security. Briefly, entities should have fixed trust models to be able to make better trust decisions based on their needs from security systems of services in open environments.

Since trust has a subjective nature, entities should have their own trust models that reflect the subjective nature in open environments. Moreover, entities should have dynamic trust models that may be changed frequently to reflect the openness of the environments. For instance, the amount of information is significant for computations of trust. Each entity may have different information gathering model related to trust computations. Moreover, these information gathering models may contain many sub-

jective factors as expected. Therefore, each entity is expected to have different models to determine the subjective factors. Moreover, the models may be changed frequently according to recent conditions of open environments. Therefore, entities should have dynamic trust models that meet requirements of trust in open environments.

In this thesis, our aim is to provide a modeling approach for formal representations of trust and formal representations of information related to trust computations based on needs of an entity from the security system of a service in open environments. The goal of the modeling approach is to formally represent trust by using both common models and different models of entities related to security systems of services in such environments. Therefore, an entity can handle both common requirements and subjectivity requirements related to modeling of trust in relation to security systems of services based on needs of the entity in open environments.

1.2. Contributions

This thesis presents a top-down approach for representing and modeling of trust based on needs of an entity related to the security system of a service. Our approach results in a hybrid modeling approach that contains two types of models, namely one core model and crust models. We call such modeling approach Core-Crust Modeling Approach (CCMA). CCMA is outlined in Figure 1.1.

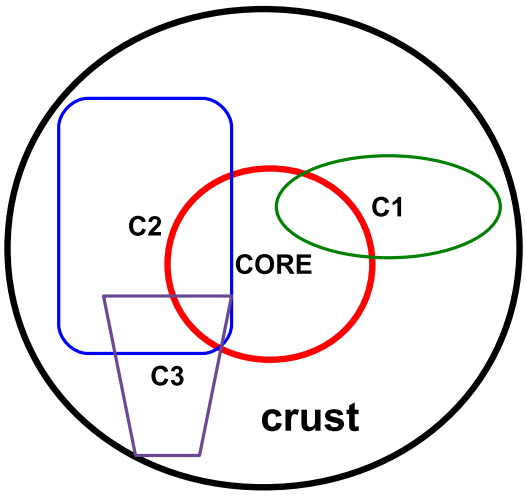

Figure 1.1. Core-Crust Modeling Approach.

In our modeling approach, entities should have a common trust model to make trust decisions related to security systems of services in open environments. We call the core model of an entity to such model related to trust computations of the security systems based on needs of the entity. Entities are expected to have same core models that may not be modified frequently. The core model represents common requirements of entities from security systems of services related to trust computations. Since the core model is a common model for all entities in open environments, the model enables entities to interact and communicate with each other related to trust computations of the security systems. CORE represents the core model in this thesis as shown Figure 1.1.

Trust has subjective property and we represent the property with crust models in CCMA. In our approach an entity may have many crust models related to security systems of services. Moreover, each entity may have different crust models related to the security systems in open environments. A crust model may be modified very frequently depending on recent needs of the entity on which the model is applied. Crust models of an entity represent specific needs of the entity from the security systems of service in open environments. In this thesis, we have three crust models as shown in Figure 1.1, which are C1, C2, and C3.

In CCMA, a crust model is connected with the core model with at least one parameter. Particularly, a crust model depends on the core model so the relation between the core model and the crust model is ensured with parameters of both models. More specifically, a crust model may be designed to determine a parameter of the core model. The connection between the core model and a crust model is represented with the intersection of the core model and the crust model. For instance, we represent the connection between CORE and crust model C1 with the intersection between CORE and C1 as shown in Figure 1.1.

A more specific crust model may be used to determine a parameter of another crust model more precisely. In this case, the crust model and the core model may not share a parameter. However, the crust model has to be connected to the core

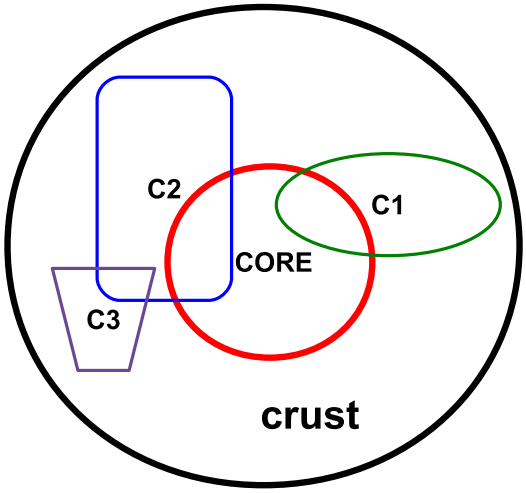

Figure 1.2. A correct example for Core-Crust Modeling Approach..

model indirectly over another crust model. For example, crust model C3 is connected to the core model over crust model C2 as shown in Figure 1.2. A crust model is not independent from the core model as shown in Figure 1.3. This modeling approach ensures an entity to have its own trust metrics related to the security system of a service and compute the metrics based on its own needs.

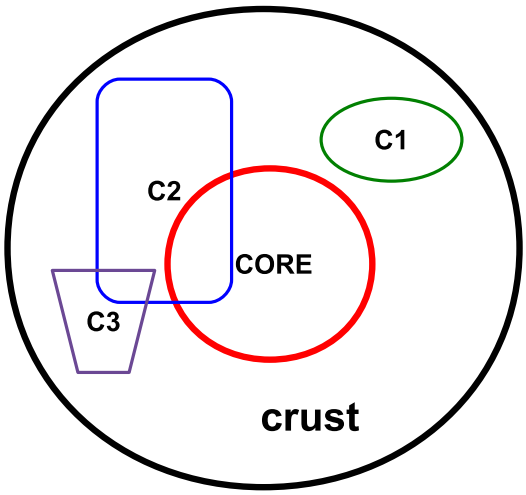

Figure 1.3. An incorrect example for Core-Crust Modeling Approach.

In this thesis, the relationships of our models are shown in Figure 1.1. CORE represents the core model of an entity in this thesis. Crust models C1 and C2 are related to obtaining information for trust computations. Crust model C3 is a more specific crust model that aims to improve security information related to trust computations. The model also describes the formal representation of a parameter related to crust model C2 in more details.

CCMA shows that our models can be improved and applied to many entities related to trust computations of security systems based on needs of an entity in open environments. Therefore, our trust modeling approach is dynamic and open for improvements. Major contributions of the thesis are as following:

- *Trust Core Model (CORE):* Trust Core Model is for assessing the trust of the security system of a service from an entity point of view. The core model contains architecture of an entity for assessing the trust of the security system in open environments. An entity may assess the trust of all properties of the security system or some properties of the security system. The trust assessments are carried out according to proposed trust assessment metrics. Additionally, we present a possible trust assessment process in an entity to assess the security system of a service based on its private needs. We support the proposed model with a case study, where the behaviors of proposed metrics are analyzed with scenarios and simulations.

- *Trust Extraction Crust Model (C1):* Trust Extraction Crust Model is a model for extracting trust information from the security system of a service. We formally represent security policies and security systems to extract trust information according to needs of an entity. The formal representation ensures an entity to extract trust information about a security property of a service and trust information about whole security system of the service. The proposed model has been applied to the security system of a management service of patients' account as a case study. The proposed model has been evaluated experimentally with simulations in the case study.

- *Information Flow Crust Model (C2):* Information Flow Crust Model is a formal model for flow of security evaluation information on entities that model is presented with a new formal model for trust assessment based on the security evaluation information. In the proposed information flow model, an entity obtains experiences directly and indirectly. Additionally, the entity receives recommendations and confidences of recommendations about the security system of

the service from other entities. Moreover, we introduce novel subjective factors related to the information flow model. The trust assessment model about the security system of a service can be applied to entities in emerging open networks. The proposed model for flow of security evaluation information and the trust assessment model have been applied to an online hotel reservation service as a case study. Two proposed models have been evaluated experimentally with simulations in the case study. Experimental evaluations of two proposed models result in more accurate trust assessment of the security system of a service in emerging networks.

- *Interoperability Crust Model (C3):* Interoperability Crust Model is a model for trust based security interoperability of services. The amount of information affects the speed and accuracy of trust computations. In our model, security systems of services are similar to the security system of a specific service. Therefore, the aim of the trust model is to increase security experiences related to a service to ensure fast and accurate trust computations. The security interoperability model is analyzed with case studies and simulations. The case studies and simulations have shown that the proposed security experience gathering model significantly improves security experiences which results in fast and accurate security interoperability decision making.

1.3. Organization of the Thesis and Publications

The rest of this thesis is organized as follows:

In Chapter 2, we provide an overview of trust and security. We illustrate the place of trust in computer security by dividing security into two sets, namely soft security and hard security. We give many definitions of trust in literature. Then, we give our definition of trust. Moreover, provide some properties of trust relationships and some trust related issues. In this chapter, we also present general trust models. Furthermore, we explain some trust related works in literature. Finally, we mention some open research problems related to trust. Briefly, the aim of this chapter is to

clarify our contributions.

In Chapter 3, we introduce our core model for trust assessment of security system of a service from entity point of view. Specifically, we provide the architecture of a trust assessment system of an entity. We show our formal model of the environment for trust computations. We formally represent security system of a service to be able to compute our novel trust assessment metrics. Moreover, we show how trust assessments are carried out according to our core model. Hospital Online Appointment Service is simulated and analyzed as a case study to show the applicability of the proposed model. Some parts of this chapter are based on [21] and [22]. Some parts of this chapter are published in [23].

Chapter 4 concentrates on the extracting trust information from security system of a service based on the needs of a specific entity. We consider a security mechanism as security policy enforcement so that we formally represent security policies and security mechanisms to be able to extract information for trust computations based on the needs of an entity. We present our model for extracting trust information as a crust model. We also provide Dental Clinic Patient Service case study to show our contributions. The case study is simulated and analyzed with two scenarios. Parts of this chapter are published in [24].

In Chapter 5, we propose a crust model for trust assessment of security system of a service based on flow of security evaluation information. The trust assessment is carried out according to the needs of an entity. In this model, there are three types of security evaluation information, namely experience, recommendation, and confidence. Security evaluation information is obtained from many entities and many services according to a specific service. Trust assessments are carried out according to the security evaluation information. Flow of security evaluation information and the trust assessment model are explained with Online Hotel Reservation Service case study. The case study is examined with simulations. Some parts of this chapter are based on [25] and [21]. Moreover, some parts of this chapter are published in [26] and [27].

In Chapter 6, we present a specific crust model for trust assessment that can be used for security interoperability decision making in convergent services. In this model, only security experiences are used to compute trust metrics. The amount of security experience related to a specific service is increased by using security experiences from other services that have similar security systems. Specifically, we introduce a security similarity ratio for security systems of two services and a formal model for the similarity ratio. Moreover, we show how the similarity ratio may be computed and used to obtain more security experiences related to the security system of a specific service. We provide case studies and simulations to prove the advantages of the proposed trust assessment model for security interoperability decision making. Simulation results show that the model provides better security interoperability results. Parts of this chapter are based on [22].

Finally, we give our concluding comments and summarize the main contributions of this thesis in Chapter 7. We also point out some future research directions related to trust assessment of security systems based on specific needs of an entity.

As mentioned above, the main contributions presented in this thesis have been submitted or published in [21–27].

2. OVERVIEW OF TRUST AND SECURITY

Nowadays, trust is regarded as a significant precondition for the adaptation of new properties and technologies in computer science. Actually, trust is quite interdisciplinary subject studied in many fields of science. Additionally, it has diverse meanings, properties, and related issues in different fields of science. However, the diversity is not good for security because it contradicts with requirements of security. Security requires clarity and precision. For example, trusted computing bases do not guarantee trust, they only ask for it [20]. Therefore, trust necessitates precise technical definition. On the other hand, the precise technical definition might actually mislead the public. Thus, trust notation in the field of computer security requires modeling of trust relationships as well as precision of trust definition.

Security is one of basic needs of all states, institutions, livings, and furthermore it is going to be essential for machines because their complexities and influences on life are continuously increasing. Usually security is described as state of being defended against undesired situations and actions such as being protected against criminals and dangers. In computer realm, the definition of security has been both expanded and changed because the interconnection of systems and their complexity has been varied. Moreover, new types of systems have been introduced. In literature, computer security includes topics such as protection of information from theft and corruption that are discussed in the categories of authentication, confidentiality, integrity, and availability. Computer security is generally defined with security policies [2, 18]. Most of the time, people enforce more security demands to computers than they are capable to cope with. The demand makes computer security more challenging and it necessitates formal representation and analysis. Formal representation and analysis of critical systems like security are broad topics studied in [11–16].

The aim of this chapter is to provide an overview of trust in relation with security in computer science. To accomplish this goal, the relationship between trust and security is presented and then trust is examined from different points of view. Moreover,

some trust based works in literature are also explained to show the applicability of trust in computer science.

The reminder of this chapter is organized as follows. Section 2.1 shows the relationships between trust and security. We present the definition of trust in different sciences in Section 2.2. Section 2.3 contains an overview of properties of trust. We examine trust related issues in Section 2.4. We explain the meaning of metric related to trust in Section 2.5. Section 2.6 describes general trust models. We present some trust based applications and models in Section 2.7. Finally, we show some open research problems related to trust in Section 2.8.

2.1. Hard Security and Soft Security

One of the most significant issues in modern computing is security. Traditionally, security mechanisms protect assets and resources from malicious users by restricting access to authorized users only. Moreover, these mechanisms allow authorized users to perform actions that are related to their task only. The term information security is used for protection of information assets. Briefly, the aim of information security is to preserve confidentiality, integrity, and availability [28]. On the other hand, networks, operating systems, software applications, and other computing systems each have their own security requirements. Therefore, the area of computer security has many security solutions. These solutions have taken the form of traditional cryptographic solutions, passwords, and access control mechanisms. However, researchers have been exploring the potential of security mechanisms that are based on some aspect of social control mechanisms with the increasing use of open systems, such as the Internet and service-oriented environments.

Traditional security mechanisms do not preserve from misleading information in open systems. In many cases, resources may mislead entities by offering wrong information or resources because of many reasons, such as contradicting interest relations between the interacting entities and resource providers. Therefore, there is uncertainty about the accuracy of provided resources and information. However, the uncertainty

is not good for traditional security mechanisms [20] so that mechanisms are unable to protect entities against such threats in open environments. This is the problem that is not addressed by traditional security mechanisms [29].

An entity can have some information about some other entities independently from any centralized authority in open environments. Specifically, entities can obtain information by interacting with each other or observing other entities. Social control mechanisms allow the entity to have such information [30].

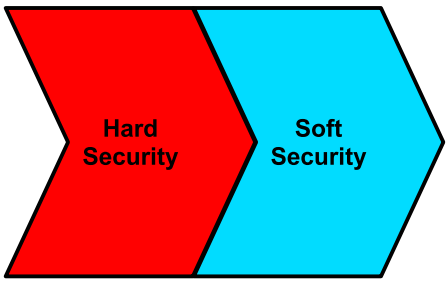

Figure 2.1. The relationship between soft security and hard security in computer science.

Social control mechanisms extend the traditional security mechanisms to cope with security problems. The extended security view is first described by Rasmussen and Jansson [30]. They used the term hard security for traditional security mechanisms and soft security for social control mechanisms. From that point of view soft security is the complement of hard security as in Figure 2.1. Trust based solutions cover both hard security and soft security mechanisms.

2.1.1. Hard Security

Hard security mechanisms provide security with mechanisms that have a common characteristic [31]. Traditional security mechanisms are classified as hard security mechanisms so the characteristics of traditional security mechanisms are also characteristics of hard security mechanisms. The use of an authentication mechanism with a user id and a password is an instance for a hard security mechanism.

In hard security, there is always a security policy whereas in soft security, there

may not be a security policy because of the lack of policy that defines the behavior [29]. Moreover, hard security mechanisms assume that components work as intended so that components cannot be used illegally [30]. However, hard security mechanisms may fail because of unexpected behaviors of some components. Therefore, hackers use the unexpected behaviors of components to reach their goals.

Systems are unprotected if hard security systems are broken. Once hard security systems are broken everything is prone to the attacks of intruders [30]. On the other hand, there are trust approaches based on hard security, such as X.509, PGP, and PolicyMaker [28]. Moreover, existing standards like WS-Security [32], WS-SecurityPolicy [33], and WS-Trust [34] ensure hard security mechanisms for applications in service oriented environments. Furthermore, hard security mechanisms are used to provide trustworthiness, such as trust management for mobile computing platforms by implementing root trust [19].

Briefly, traditional security mechanisms are considered as hard security mechanisms. In literature, the term hard security is used together with the term soft security [35–37]. Hard security mechanisms are convenient in case of information completeness and correct behavior of all components. However, current trend is to use both hard security and soft security mechanisms together to ensure higher security.

2.1.2. Soft Security

Soft security approach considers that malicious intruders may access to the system. The idea behind soft security is that the intruders may be identified and prevented from harming the others in the system [30]. Social control works in that way so it is considered as a soft security mechanism. Entities in open systems like Internet may apply social control mechanisms as a soft security to improve their security.

Hard security aims to protect confidentiality, integrity and availability properties of assets in a system. On the other hand, the goal of soft security is to improve the quality of a community by using ethical behavior of the community [29]. Soft security

mechanisms use collaborative methods to identify and sanction members of a community who do not obey ethical norms of the community [28]. Soft security mechanisms assess the behaviors of members against the ethical norms [29]. In electronic world, a problem is that what constitutes the ethical norms.

The soft security approach is convenient for systems in open environments [31]. In such environments, the security is not centralized and it depends on participating entities. Moreover, there is no concrete security claims to prevent security breaches. Therefore, the security depends on participants. For these reasons, soft security mechanisms are more robust than hard security mechanisms [30]. For example, if an intruder passes the hard security mechanisms of a system, soft security mechanisms are the last defense line of the system [37].

Social control mechanisms are soft security mechanisms that related to the behavior [28–30, 35–38]. Soft security mechanisms allow entities to interact as long as they behave honestly and nicely. Soft security mechanisms may be used after hard security mechanisms as the last defense to protect the system. Therefore, soft security mechanisms can be considered as the complement of hard security mechanisms to ensure more secure system.

2.2. Defining Trust

Trust is investigated in different fields of science and the definition of trust highly depends on the context and the research field. On the other hand, there is no agreement about the definition of trust [1, 20, 39, 40].

2.2.1. Trust Definitions in Social Sciences

The concept of trust is initially used in social sciences, such as sociology and philosophy. Sociological trust is defined as a particular level of the subjective probability of an agent that assesses a future action of another agent in [41]. Subjective factors, risk, confidence, and security are used in the definition of trust, where trust shows the

expected behavior of an entity [4, 42–44]. For instance, the trust definition of Deutsch points out that trust is subjective for an action under foregoing conditions that vary for each person in different situation. This definition is applicable to relations of people in everyday life of social environment. Specifically, the trust definition of Deutsch is as [43]:

"An individual may be said to have trust in the occurrence of an event if he expects its occurrence and his expectation leads to behavior which he perceives to have greater negative motivational consequences if the expectation is not confirmed, an positive motivational consequences if it is confirmed ".

In practical world, the trust definition of Gambetta is one of the most influential works. The definition of trust emphasizes that trust is related to belief and estimation for performing a future action in a specific context as [41].

"When we say we trust someone or that someone is trustworthy, we implicitly mean that the probability that he will perform an action that is beneficial or at least not detrimental to us is high enough for us to consider engaging in some form of cooperation with him. Correspondingly, when we say that someone is untrustworthy, we imply that that probability is low enough for us to refrain from doing so."

Mayer *et al.* extend the trust definition of Gambetta by adding vulnerability associated with risk that one is willing to take as [45]:

"The willingness of a party to be vulnerable to the actions of another party based on the expectation that other will perform a particular action important to the trustor, irrespective of the ability to monitor or control that other party."

2.2.2. Trust Definitions in Computer Science

Despite the nature of trust is subjective, the concept of trust has been widely used in many contexts of computer science because it provides diverse decision making

options in different circumstances. Similar to social sciences, trust is defined in different manner in computer science [39, 46–49].

Paolo Massa defines trust in real online systems as the judgment expressed by one user about another user, often directly and explicitly, sometimes indirectly through an evaluation of the artifacts produced by that user or her activity on the system [1]. Massa also enlightens the definition of trust by identifying categories of trust in online systems and surveys them in which systems can be grouped according to their similar properties and common features. Some of these categories are e-marketplaces like eBay, opinions and sharing sites like Last.fm, business and job networking sites, social entertainment sites, and so on.

Another definition and survey related to online systems is proposed by Josang *et al.* [46], [50] who defines two general trust notations, namely reliability trust and decision trust. Reliability trust is the subjective probability by which an individual expects that another individual performs a given action on which its welfare depends on. Decision trust is related to a feeling of relative security that affects decision making. Josang *et al.* also have other notations of trust such as identity trust. The common property of all these definitions is that they are related to reputation. The reliability trust definition does not address the situation when it is possible that the damage is too high to choose the most reliable trust branch [51].

Mui *et al.* developed a mathematical model to compute probability of trust based on existing experiences for predicting future behavior of an agent. They also provide a trust definition regarding reputation as [52]:

"Trust is a subjective expectation an agent has about another's future behavior based on the history of their encounters."

Grandison and Sloman define trust for Internet applications that definition contains a composition of many different attributes like reliability, dependability, honesty, truthfulness, security, competence, and timelines in a specified context [4]. Trust defi-

nition of Grandison and Sloman is developed after analysis of existing trust definitions. They create their own trust definition as follows:

"Trust is the firm belief in the competence of an entity to act dependably, securely, and reliably within a specified context."

Donna Andert *et al.* pay attention to formalizing trust and analyzing risk for security architecture of all business systems to ensure that they are protected according to their stated requirements and identified risk threshold. Their definition of trust is based on the ITU-T X.509 document, where behavior of one entity depends on the assumptions and expectations of second entity [17]. More specifically, they state that trust is a kind of binary relation between peers or a set of compounded binary relationships based on validation of unique individual identity. They define spontaneous trust that can exist without original entity authentication and bootstrapping. Donna Andert *et al.* also offer set of principles for defining trust and trust modeling for security architecture.

Trust is defined as the possession of authentic and valid credentials necessary for access control at an end point in the context of access control [53]. There are also other definitions related to trust in [53], such as trust negotiation, dynamic trust negotiation, circle of trust, and trust contract. Specifically, circle of trust is defined as a network of locally trusted intermediary peers that a peer or an entity trust, collaborates with through one or more trust contracts between each peer in which a trust contract is an agreement that exists between two entities. On the other hand, trust is defined by using role mapping for virtual organizations such as cross-domain role mapping and forbidden role mapping [7].

Yan Lindsay Sun *et al.* define trust for distributed networks to cope with trust management vulnerabilities as a bridge between social needs and security solutions [18]. The role of trust is divided into three titles that are prediction and diagnosis, simplification and abstraction, and integrating social needs into design. For instance, prediction and integration solves the following four important problems; assistance in

decision making to improve security and robustness, adaptation to risk, misbehavior detection, and quantitative assessment of system level security properties.

Blaze *et al.* describe aspects of decentralized trust management [54]. They refer all components of network services as the trust management problem. The trust management problem is first defined by Blaze *et al.* Their approach to trust management is based on unified mechanism, flexibility, and separation of mechanism from policy. Unified mechanism provides a common language for policies, credentials, and relationships with which we can make it possible for network applications to handle security in a comprehensive, consistent, and largely transparent manner. On the other hand, by supporting local control of trust relationships, the need for assumption of a globally known, monolithic hierarchy of certifying authorities is avoided.

There are definitions of trust for computing environments such as service oriented computing, computing platforms, and ubiquitous computing. Denis Treck gives informal and formal definitions of trust for service oriented trust modeling, qualitative trust modeling, in contemporarily computing environments. Manifestation of reasoning and judgment processes is described as informal definition of trust that has origins in psychology [55], whereas, formal definition of trust is a relationship between agents A and B, which may be described as trusted, untrusted or undecided. Karl Krukow gives definitions of trust for global ubiquitous computer according to trust models, trust structures, and trust management approaches such as credential-based trust management and experience based trust models [5]. Another definition of trust is related to ubiquitous environments presented by Walter *et al.* [56]. Trust means an indication that an entity fulfills its commitments, for example, a device does not reveal confidential data. They also define trust certificates as evidence on the trustworthiness of devices. On the other hand, in knowledge management systems, trust is a composition of many different attributes that are reliability, dependability, honesty, truthfulness, security, competence, and timeliness, which may be considered depending on the environment in which trust is being specified [40].

Zheng Yan proposes trust management for mobile computing platforms, such as

mobile phones, where there are two kinds of trust, soft trust and hard trust [19]. The soft trust solution provides trust based on trust evaluation according to subjective trust standards, facts from previous experiences and history. On the other hand, hard trust solution builds up trust through structural and objective regulations, standards, as well as widely accepted rules, mechanisms and sound technologies such as PKI and Trusted Computing Platform. Hard trust provides guarantee for the soft trust solution to ensure the integrity of its functionality. Soft trust can provide a guideline to determine which hard trust mechanisms should be applied and at which moment. It provides intelligence for selecting a suitable hard trust solution.

Ugur Kunter and Jennifer Golbeck analyze social trust from a computational perspective in a relation between the term confidence and the term trust in [48]. They describe an approach that gives an explicit probabilistic interpretation for confidence in social networks and presents a new trust inference algorithm, SUNNY. The algorithm uses a probabilistic sampling technique to estimate confidence in the trust information from some designated sources, where trust is represented by a directed graph, trust graph.

Tatyana Ryutov presents a policy specification approach in [49] that combines social sciences and trust theory to facilitate ad-hoc interactions of self-interested parties in open environments and that contains definitions related to trust. Her socio-cognitive approach allows to reason about uncertainty and risks involved in a transaction, and automatically calculate the minimum trust threshold needed to mitigate the vulnerabilities. The trust threshold comprises the core of security policies that govern the interactions. She defines trust as a decision to accept vulnerability, participate in an exchange, with expectation of positive or negative outcome of an exchange which depends on the actions of the opponent. In addition, she divides trust to subjective trust and objective trust. Subjective trust encompasses trusting attitude due to perceived qualities or abilities, e.g. reputation, skills, and profiles, of the other participant. Objective trust means that one has formed an intention to trust, participate in exchange, regardless of beliefs about the qualities of the other party due to any or all of the following reasons. It is economically not profitable for the other party to cheat because

of severe penalties in the case of non compliance. If trust is misplaced, some compensation mechanism will mitigate the vulnerability. In other words, subjective and objective trust types are complementary to each other.

Traditional trust relations have been generally evolved slowly with time [39]. This indicates that existing trust notions are entity centric and slow to change. However, several emerging mobile networking systems are heavily, if not entirely, data centric in their functionality and operate in ephemeral environments. Raya *et al.* propose data centric trust establishment that takes dynamic factors into account such as location and time. Data centric trust changes frequently and does not steam from a single source of data. Similarly, there is a trust definition in [57] that is based on certain activities at a given time.

2.2.3. Our Definition of Trust

While diverse number of definitions of trust in computer science exists, it may lead us to confusion about trust relationships in the context of security that is an undesired circumstance [20]. In order to compare trust relationships, there is a need for precise definition of trust. It is also significant to determine the boundaries of trust both what does it mean and what does not it mean to meet the basic requirements of security.

We define *trust* as the security expectation of an entity from a service according to available security evaluation information of that entity.

2.3. Some Properties of Trust Relationships

Trust has various definitions, as well as it also has lots of properties in which some of them depend on the definition of trust. Two entities may have one-to-one trust relationship between them. However, the relationship may not be symmetric, where trust is directed. In addition, trust relationships may be one-to-many, many-to-many, or many-to-one in which groups of entities are involved in trust relation [4]. If

trust information about one entity is delegated from one entity to another conditionally or unconditionally in a group, the group uses the transferable property of trust that is generally realized by trust networks [19].

Subjectivity is another property of trust that depends on personal opinion. Actually, personal opinions are based on some factors or evidence. Almost always trust is used in particular context. Therefore, trust is context dependent.

In general, trust values are used to represent different degrees of trust that an entity may have in another. Values of trust represent the measurability property of trust. This property ensures us that trust based systems can be modeled and analyzed. One of the most significant properties of trust is dynamism, such as trust values may depend on time, actions, events, and etc. To cope with dynamic property of trust, trust management systems may use learning and reasoning solutions [19]. There are also other properties of trust like history dependency. To sum up, it is obvious that the number of trust properties varies from one system to another one. Moreover, there are many properties that trust may satisfy in literature.

2.4. Some Trust Related Issues

Since trust may have different definitions in each context, many related terms are used with different trust models and trust based solutions. In this section, we present some of the most known terms that are used to define and model trust. Specifically, we explain the terms reputation, confidence, and federation in this section.

2.4.1. Reputation

Reputation is an important concept for systems that consider trust. For instance, some online systems use reputation based trust systems. Paolo Massa's approach to reputation in online systems is that it is a sort of status that gives additional power and capability. Moreover, reputation can even be considered a sort of currency [1]. He also sees reputation as a global, collective measure of trustworthiness based on trust

statements from all the members of the community. *"Reputation is what is generally said or believed about a person's or thing's character or standing."* is the definition of reputation for online systems given by both Josang in [46, 50] and Massa in [1].

There are other definitions of reputation in the field of computer science. For instance, F. Oliviero *et al.* use the term multidimensional reputation for an autonomic approach to network security based on multidimensional trustworthiness, where reputation is a peer's global trust rating [58]. Similarly, Raya *et al.* give a definition for reputation systems on data centric trust establishment systems in [39], where reputation systems are used to monitor node actions over several interactions to compute node trust value. Zheng Yan has reputation and reputation related definitions in [19] for trust management for mobile computing platforms. Yan defines reputation as *"A public assessment of the trustee considering its earlier behavior"*. Yan also defines reputation schemes that are classified in two different categories, global reputation and local reputation, depending on the sort of reputation. Global reputation is the aggregation of all available assessments by other entities that have had interactions with a particular entity. Local reputation is based on each entity's own assessment according to past interaction with a particular entity. Thus, global reputation is a many to one relationship, whereas local reputation is one to one relationship. There is a definition of reputation for trust systems in multi-agent systems [59]. This reputation is considered as a measurement of the amount of trust one agent holds for another that is an estimation of risk versus reward for trusting another agent. Actually, there are a wide variety of reputation systems ranging from practical systems implemented in commercial applications to theoretical work [49, 57].

Traditional reputation systems make some kind of abstraction on the direct data. Actually, abstractions have some advantages, such as it decreases the need for space to store data. However, abstract representations cause lost of information. Therefore, it is impossible to verify properties of past behavior given only with the reputation information. Karl Krukow proposes a logical framework for reputation systems to cope with this problem [5]. In his framework, a declarative policy language is used to specify requirement on past behaviors, and the ground of making decisions about future

interaction becomes the verification of a future history with respect to a policy. Krukow defines a concrete reputation system to get a form of provable security guarantee as follos: *"If principle p gains access to resource r at time t, then the past behavior of p until time t satisfies desired requirement of p"*. The reputation definition of Krukow has an exact semantic that provides a way to specify security properties formally.

Most of trust based systems use reputation by taking into account reputation scores that are aggregated ratings about a given party. In other words, reputation scores can refer to a probability measure that indicates a particular agent's expected behavior in future transactions [60]. The architecture of a reputation system determines the way of collecting and distributing reputation scores. Specifically, there are two architectures to manage reputation scores, centralized and distributed architectures. In centralized architectures, a reputation center collects all ratings from participants. Then, the reputation center calculates a reputation score for every participant. Finally, it makes reputation scores available to all participants [46]. For example, the Feedback Forum of eBay uses a centralized reputation system that collects all ratings and computes scores. On the other hand, in distributed systems, there is no any central authority that manages scores. Instead, ratings and individual opinions of participant about other participants are stored in different places. When anyone needs reputation information about a particular participant, the participant obtains information from available reputation centers. The reputation system of Napster is an example for distributed reputation architecture. Additionally, reputation scores are specific to particular time interval [50]. Furthermore, there are different ways to compute reputation scores, such as simple summation or average ratings, binary ratings that are used in Bayesian systems.

Actually, reputation systems manage information about the past behavior of principals. Therefore, reputation systems may serve as a trust enabling or trust informing technology [5]. Reputation systems sometimes are called collaborative sanctioning systems to reflect their collaborative nature. In these systems, poor service providers are sanctioned to give an incentive for them to ensure quality of services. In addition, reputation systems are related to collaborative filtering systems, where different people

have different tastes, and rate things differently according to subjective tastes [46, 50]. Reputation systems are already being used in successful commercial online applications. For instance, Google uses PageRank algorithm to calculate reputation scores online [61].

While trust related systems and reputation systems are mentioned within same context in literature, actually they differ from each other. Generally, trust systems use reputation systems. Josang *et al.* propose three main differences between trust and reputation systems [46]. First, trust systems provide a score that reflects the relaying party's subjective view of an entity's trustworthiness, whereas reputation systems produce an entity's public reputation score as seen by whole community. Second, transitivity is an explicit component in trust based systems; however reputation systems usually take transitivity into account. Last, trust systems usually take subjective and general measures of trust as an input, whereas ratings about specific events are used as an input in reputation systems.

Trust and reputation relationships related to security gain more attention in contemporary researches in computer science literature. Formal modeling of reputation systems in relation with trust needs considerable work to ensure more concrete security.

2.4.2. Confidence

Like trust, confidence is subjective that is described as a state of being certain or emotional state of mind. The relationship between trust and confidence is given in [17], where trust is the source of confidence that something will or will not occur in a predictable or promised manner. Grandison and Sloman describe confidence trust for online systems [4] as *"trustor has confidence in the standard of service provided by a service provider"*. On the other hand, Yan [19] takes confidence into account as the accuracy or quality of trust where high confidence is more useful in making trust decisions.

Kunter and Golbeck propose an algorithm [48], SUNNY, for trust inference based

on probabilistic confidence model for social networks. Their claim is that SUNNY is the first trust inference algorithm that includes a confidence measure in its computation. They define confidence as *"an entity A has in another entity B as A's belief on the correctness of information provided by B"*. They also model the confidence of an entity A for another entity B as the conditional probability $P(A|B)$, where given that B transmits some information to B, the probability that A believes in the correctness of that information is $P(A|B)$. There are also other researches that use confidence and trust in literature related to security [39, 57, 58].

In security realm, identification, authentication, accountability, authorization, and availability are providers of confidence. Generally, there is a need for a satisfactory level of confidence to set up a trust relationship in attributes provided by an entity. Specifically, confidence necessitates a binding of unique attribute to a unique identity, and the binding must be able to be tested satisfactorily by a relaying entity [17]. After you reach a satisfactory level of confidence for the attributes provided by an entity, you can establish a trust relationship. Each trust model has different confidence level. For instance, direct trust model ensues high level of confidence in every entity associated with trust implementation, whereas, spontaneous trust model does not provide any level of confidence.

Confidence in relation with trust is used as a confidence level that helps to use statistical properties of trust. In statistics, a confidence level is generally described as a confidence interval or confidence bound that is an interval estimate of a population parameter. Specifically, reliability of an estimate is represented by confidence intervals. The confidence interval for a population parameter is calculated from a random sample of an underlying population for a given confidence level.

2.4.3. Federation

In computer science, the term federation is used with collaborative networks, such as devices forming collaborative networks. In these networks, capability and trust help to select roles of nodes a priori or randomly [56]. Federation based trust defines trust

relationships through cross-domain identity or attribute mapping, which is generally used in grid like secure virtual organizations. New collaborators may join and some existing ones may depart any time without disclosing all of their security policies for purposes of privacy protection [7]. In this approach, federation is the complement of centralized authority approach.

Walter *et al.* define a federation as *"a network of selective application functionality in which capabilities can be software elements installed on devices"* [56]. More specifically, federation is an issue of relationships between communicating nodes, whether the inter-nodal connectivity is represented by a client/server environment like a systems of personal device peers. Thus, trusted computing platform is extended beyond a single entity to a distributed topology by the concept federation.

2.5. Trust Metrics

We need metrics related to trust to predict acceptable outcomes for future interactions of entities that are involved in trust computation [49]. Massa defines a trust metric as *"an algorithm that uses the information of the trust network in order to predict the trustworthiness of unknown users, where a trust network is the graph obtained by aggregating all the trust statements issued by users"* [1]. Massa also divides trust metrics into local trust metrics and global trust metrics. Global trust metrics are used to predict the same value of trustworthiness of a particular user for every user, such as PageRank in which global trust metric for Microsoft is 9/10 for all users. On the other hand, local trust metrics predict the trustworthiness of other users from their point of view of every single different user. In addition, local trust metrics are personalized metrics, for instance, PageRank would predict different trust values for web page of Microsoft users who have different trust levels to different operating systems.

Local trust metrics may solve multiple identity problems in which the activity of fake identities not reached by trust propagation does not affect the active user. The reason is that local trust metrics consider only trusted users. Massa defines trust propagation for social and entertainment sites as *"other users can search through friends'*

lists of their friends and discover and be introduced to new people that may be more interesting than a random stranger" [1].

While the importance of trust as a security tool in real life increases day by day, some performance analysis of trust models must appear. Furthermore, we need performance metrics of trust models to perform precise analysis. In literature, there are various trust related metrics but the unification of metrics is difficult [59]. Therefore, finding scalable trust metrics is an important challenge, especially for local trust metrics [1]. Local trust metrics must be time efficient to predict trustworthiness of unknown peers in a short time.

Sun *et al.* introduce a trust metric called malicious node detection performance, which represents the average detection rate to cope with bad mouthing attack in distributed networks that use trust management systems [18]. Moreover, they propose another trust metric that describes the false alarm rates.

Josang *et al.* have a trust and belief metric called opinion, which expresses the relaying party A's belief in the truth of statement x [46]. In this metric, belief, disbelief, uncertainty, and the relative atomicity that represents the base rate probability in the absence of evidence are used to calculate the metric.

Raya *et al.* define trust metrics such as time freshness and location relevance for multiple pieces of evidence for data centric trust establishment in ephemeral ad-hoc networks [39]. As a final note, local trust metrics must be run one time for every single user propagating trust from her point of view, whereas global trust metrics are run only once for the entire community [1].

2.6. General Trust Models

Generally, trust models determine the degree or level of trust relationship between two entities, such as how to calculate indirect (transitive) trust between entity A and entity Y from trust propagation paths [18]. Andert *et al.* define trust modeling as

"the process to define complementary threat profile for a security architecture based on an acceptable trust model" [17]. More specifically, they define trust modeling as determining a trust model that identifies the specific mechanisms necessary to respond to a specific threat profile. Trust modeling must include implicit or explicit validation of an entity's identity or the characteristics that are necessary for particular event or transaction to occur. In some trust researches, the meaning of trust type is used instead of trust model [18].

The first trust model is the direct trust model like recommendation trust [18]. In this model, the validation of an entity's credentials is ensured without reliance on any other entity [17]. Trust is established through observations on whether the previous interactions between peers are successful. There is no propagation of trust, because all relaying parties are ingredients of the trust hierarchy and all of them contribute to trust decision in accordance with their significance. The advantage of this model is that the validation of credentials is realized by same entity without delegation of trust. Therefore, a high level of confidence is ensured in every entity associated with the trust implementation. In contrast, the disadvantage of direct trust is that it may be more labor intensive and more expensive than other trust models [17]. PKI is an example for direct trust model. In PKI, the root certificate authority initiates all trust relationships and it is the common trust entity that performs all original entity authentications and the generations of credentials that are bound to specific entities.

Second trust model is the transitive trust model in which trust is transmitted through other parties [17, 18]. Transitive trust model is also referenced as indirect trust in literature [5, 18, 19]. Transitivity property of trust is based on the delegation or propagation of trust. For instance, entity A has direct trust to entity B, and entity B has direct trust to entity C, therefore entity A has transitive trust to entity C but entity A does not need to validate trust to entity C [17, 46].

Actually, trust is not transitive in some situations [4, 54, 60]. For example, trust may be transitive in one context for specific entities however same entities may not use the trust information gained in the specified context into another context to derive

transitive trust for these entities.

There are two factors that must be handled for trust transitivity. First factor is how and when to collect trust information. Second factor is how to calculate trust values for trust propagation [18]. From security point of view, the advantage of transitivity is that it enables us to connect different entities that share similar security policies therefore the effort needed to validate the credentials is decreased [17]. Transitive trust model is common in distributive or peer-to-peer systems.

Assumptive trust is the last trust model that is the formal name of spontaneous trust. This trust model does not necessitate any validation process [17]. The pretty good privacy (PGP) web of trust is this kind of trust model. The difference between transitive trust model and assumptive model is the validation process. In transitive trust model one requires a validation process and in assumptive trust model one does not need any validation.

2.7. Trust Research in Computer Science

Trust and trust related problems have been investigated over years and still attracting academicians as an emerging research field. In this section, we present some trust related research in computer science. Particularly, we concentrate on applications of trust related to modeling of trust, formal representations of trust, trust and security, and trust in networks.

One of the earliest research related to trust and security is proposed by Matt Blaze *et al.* [54]. The research is starting point for many contemporary researches related to trust and security in distributed systems. Authors claim that a coherent intellectual framework is needed for the study of security policies, security credentials, and trust relationships. All components of network services are referred as the trust management problem. Aspects of the trust management problem include formulating security policies and security credentials, determining whether particular sets of credentials satisfy the relevant policies, and transferring trust to third parties. Trust

management is based on unified mechanism, flexibility, and separation of mechanism from policy. The research contains a trust management application, namely PolicyMaker, which is one of the earliest trust management applications. PolicyMaker binds public keys to predicates. The successor of PolicyMaker is KeyNote.

Another research of Blaze *et al.* is presented in [62]. The research is complementary work of research in [54]. A new policy evaluation approach is proposed, which is mentioned as a complementary solution to static-verification techniques in [62]. Specifically, the problem is that traditional decentralized trust management architectures do not directly address questions such as policy changes under rapidly changing network conditions or revocation and autonomous versus centralized control. Fine-grained access control is possible by using a formal specification of policy with a policy language that can be understand by both managers and interconnected systems. Trust management provides the basis for communicating policy among system elements. In dynamic trust management, specification and enforcement of a security policy for a given service are decomposed according to the service's structure and partially conferred on the secondary services in terms of their policies. Modern service-oriented architectures require a policy specification that should be dynamic to accommodate changes in both the system and its environment. Cooperative policy evaluation is considered as a starting point for dynamic trust management, where a global policy controls evaluation of trust levels for principals in the system.

An expanded trust model for distributed system is proposed in [63]. A thorough understanding of both the psychological and engineering aspects of trust is necessary to develop an appropriate trust model. In conventional trust models, trust is formed from security, usability, reliability, privacy, availability, and safety. In the expanded trust model, trust is still a composition of aspects of conventional models. Moreover, trust aspects are composed from verification, knowledge, experience, and trust propagation. The expanded trust model is explained with two application systems, which are electronic voting system and inter-vehicle communication.

Security vulnerabilities are significant for trust establishment in distributed net-

works. The benefits of trust in distributed networks, the vulnerabilities in trust establishment methods, and the defense mechanisms are investigated in [18]. The role of trust is divided into three categories that are prediction and diagnosis, simplification and abstraction, and integrating social needs into design. Specifically, prediction and integration solves the following four important problems; assistance in decision making to improve security and robustness, adaptation to risk, misbehavior detection, and quantitative assessment of system level security properties. Additionally, some attacks against trust establishment methods are identified and defense techniques are developed. For instance, bad mounting attack is described as when recommendations are taken into consideration, malicious parties can provide dishonest recommendations to frame good parties and/or boost trust values of malicious peers.

A development framework related to security and trust issues in ubiquitous environments is presented in [56]. The framework combines security policies, certificates and an enforcement protocol as a solution to provide security and trust in ubiquitous applications and services. Security policies define the constraints when, how and which mobile devices can be use in a mobile business application. Security implies protecting corporate resources against threats and attacks. Trust means as an indication that an entity fulfills its commitments and it implies an availability of user and service credentials.

Virtualized trust computing platform for adaptive security enforcement of web services interactions is proposed in [64]. A method for combining software resource level security features with the hardware-based security mechanisms and system virtualization approaches is described. The method contains trust-based architecture for protecting the enforcement middleware deployed at the policy enforcement endpoints of web and grid services. The aim of the research is to secure the execution environment of the applications by providing virtual machine level separation that maps from logical domains imposed by web services level enforcement policies.

Trust based protection of software component users and a designer is studied in [65]. Two security problems are defined for component based software design methods.

The first problem is security risk in software based design method that originates from the software component where a malicious component may attack the application incorporating with it. The second problem is risk that originates from the owner of the application in which he may incriminate a component designer falsely for any damage in his application in reality was caused by somebody else. A trust based solution is presented to cope with the second problem. A trust information service is used to store trust values of the component user to prevent the component user to send wrong reports. Wrong reports result in a wrong trust value of the component.

An automatic approach to network security based on multidimensional trustworthiness is proposed in [58]. The goal of the approach is to improve network security through manipulating and combining data coming from multiple sources. To do this, a four layered model is presented, namely REFACING (Relationship-FAmilarity-Confidence-INteGrity). REFACING is used for dynamically renewing network nodes' reputation. Each layer provides different kind of information to manage intrusion detection system that information is based on reputation. For instance, the relationship layer provides information about the existence of some form of connections among detection components, such as probes, detection engines, etc.

Trust and mistrust in secure applications is investigating in [66]. Specifically, the effects of the common trust assumptions are analyzed, such as why these assumptions are often wrong, how trust assumptions can arise during an application's development process, and how to minimize the number of problematic trust assumptions in an application. Generally, software developers misunderstand trust or ignore trust related issues. In most projects, dangerous trust assumptions come from two major areas: incomplete requirements and miscommunication between development groups. Actually, faulty trust assumptions can be made at any point in the software development process. As little trust as possible should be placed in the hands of external components. The software under development must be written to handle malicious activities by any and all entities. Authors claim that current software development process is an ad hoc fashion. If developers believe a component will run in a trusted environment, they may not utilize proper secure coding practices. In order to increase one's confidence in

the security of the application, the application should minimize the amount of trust it places in its execution environment.

In [67], security and trust through electronic social network based interactions are analyzed. The motivation is the complexity of establishing trust in the Web of Trust. An electronic social network model is used to solve the problem. The model consists of a trust establishment mechanism and a trust management system. In this model, trust establishment consists of two steps, trust assessment and trust declaration. Trust management is based on dynamically determining trust levels and describing trust propagation. The model consists of two trust levels, marginal and full trust levels, where the second one is strongest one. The trust propagation enables users to benefit not only from their direct trust relations but also from a larger network of people they might gain confidence in. Another research related to social networks is in [68], where Internet social network communities are investigated according to risk taking, trust, and privacy concerns.

There are many trust researches in relation to security in computer science. For instance, achieving privacy in trust negotiations with an ontology-based approach is proposed in [69]. Security threats scenarios in trust and reputation models for distributed systems are investigated in [70]. Security as a service in relation to trust is investigated by Kaufman [71]. In another research, trust and risk are used in role-based access control policies [72].

Recently, formal modeling of trust in relation to security has become an important issue. In [5], Krukow investigates theories of computational trust. The research is a collection of various methods and tools related to formal representation of trust management. It surveys some important trust management methods and mathematical modeling approaches. The research contains a secure trust model.

Trust network is analyzed with subjective logic [60]. Subjective logic [73] provides a simple notation for expressing transitive trust relationships and defines a method for simplifying complex trust networks. Therefore, the networks can be expressed in a

concise form and they can be computationally analyzed. Trust networks are directed graphs represented as canonical expressions. Authors use and specify trust transitivity to analyze trust networks. The scope of transitive trust is specified and formulated.

An interoperable context sensitive model of trust is proposed in [74]. The model allows formalizing trust relationships between a trustor and a trustee. The trust relationships depend on the context that is represented with context graphs. Trust relationships are analyzed within a context and between different contexts. Formalizing trust relationships between related contexts allows predicting trust relationships with incomplete information. A context is described by a set of keywords such as experience, knowledge, and recommendation. Multiple contexts may be related using generalization/specification relationships or composition relationships by generating context graphs. Context graphs may not be merged because they may contain conflicting information.

UML sequence diagrams are extended to model trust-dependent behavior with the aim to support risk analysis in [75]. The research adds some properties to UML sequence diagrams to represent some trust relationships. A language to formalize trust in these diagrams is proposed. The language allows trust dependent behaviors to be described at a level of abstraction that is suitable for communication between different groups of stakeholders in a risk analysis situation.

Trust is modeled with fuzzy computational models. For example, fuzzy computational models for trust and reputation systems are proposed in [76]. The research contains an analysis for both the trust model and the reputation model on a movie recommendation system by experiments. The experiments results show that incorporating trust and reputation concepts separately gives a two level filtering methodology to enhance the recommendation accuracy through reputation-based similarity and trust-based filtering. Briefly, the research is about modeling trust and reputation form a fuzzy window.

A trust model with uncertainty quantification and reasoning for pervasive com-

puting is presented in [77]. The trust model is based on cloud theory. In the model, trust relationships between entities are formally represented. The model also contains an algorithm to calculate trust propagations and trust aggregations for reasoning in pervasive computing environments. Uncertainty is emphasized in trust relationships between entities, where the relationships are fuzzy and stochastic.

In [78], Shi *et al.* propose a trust model with statistical foundation. The goal of a trust model is to assist users with decision-making in online interactions by using past behavior as a predictor of likely future behavior. Decision making helps us to make optimal choice in online interactions. Authors use stochastic models of web services in the trust model. They also categorize trust representations. For instance, qualitative trust representation is sufficiently rich to specify a difference in characteristics between slow and fast dynamics, but it is not rich enough to specify more subtle differences in characteristics. On the other hand, in a quantitative trust representation, trust is measured by a continuous real value, which bounded between lower and upper limits. The proposed model is based on an abstract model of the trusted entity. Therefore, it is important to identify the space of possible outcomes, which determines the nature of the associated trust model. The basic assumption is that an entity behaves like a stochastic process and the model is constructed by considering this assumption.

A socially inspired reputation model is proposed in [79]. The model is inspired by the phenomenon of reputation in human societies that is proposed for enabling service consumers and producers to be mutually aware of their trustworthiness. Precise formal definitions of trust and reputation are presented in the research. Trust is formally defined and the properties of trust are specified, such as reflexivity.

In [80], Marsh investigates formalizing trust as a computational concept. He presents detailed investigation of trust in social sciences and discusses them by considering computational aspects to provide tool for decision making. The formalization can be used in artificial agents for trust based decision making.

Trustworthy web services provisioning for differentiated customer services is pro-

posed in [81]. Resource provisioning problem for web services is defined and the problem is solved with a trust-based approach subject to constraints of trustworthiness, a percentile response time and availability. The problem is a resource management problem for web services under service level agreement guarantee. The service level agreement combines several qualities of services like security and performance. A framework is proposed to solve the problem.

Trust negotiation for web services is modeled a state-machine based modeling approach [82]. Trust-Serv framework is proposed that supports life-cycle policy management and automated policy enforcement. Trust-Serv models are independent from the formal language used to enforce the policy. The majority of users are unknown to the provider. Among access control models that address the problem of unknown users, trust negotiation has attracted the most attention. Trust negotiation policies describe which credentials are required to access some resource and which can be disclosed during the negotiation. Because developing trust negotiation policies is generally considered to be difficult, providing good modeling tools is important for successful trust negotiations.

Building trustworthy software agents is presented in [83]. According to the research, it is not enough to assume that well-designed software agents will provide security and privacy users needed. A model is presented to determine agent acceptance in which feeling of trust and perceptions of risk is combined in opposite directions to determine a user's final acceptance of an agent technology. People need to trust agents related to their private information. Moreover, the agents have to handle that information in a secure fashion. Two factors affect the success of an agent. Trust contributes positively while perceived risk contributes negatively for success of a software agent.

The use of complex algorithms and powerful fixed infrastructure is infeasible due to the nature of pervasive environment and tiny pervasive devices. A trust based secure service discovery model for truly pervasive environment is presented to cope with this problem [84].

Building robust and effective reputation systems for use in decentralized autonomous networks, especially in peer-to-peer networks is investigated in [85]. Peer-to-peer reputation system research is conducted under the assumption that all peers in the networks are unknown and untrusted. Game theoretic approach and economic analysis method are used to investigate properties of reputation systems and trust. Formal models are proposed and then analytical analysis of reputation systems is studied. Some simulations related to these analyses are carried on for verification of such analytical analyses. The thesis also contains several performance metrics for evaluating reputation systems.

Trust modeling research has increased in computer science. For instance, managing trust is significant for P2P computing, particularly in critical areas such as e-commerce. Probabilistic estimation and social networks are compared for managing P2P reputation [86]. Design, analysis, and deployment of omnipresent formal trust model with trust bootstrapping for pervasive environments is proposed in [87]. Bayesian network based trust management for secure collaboration in uncertain environments is investigated in [88]. Modeling initial and repeat online trust is important in B2C E-commerce [89]. Another research related to trust models investigates how agents can handle unfair third-party testimonies in computational trust models [90].

Trust based solutions have been widely used in networks. For instance, the EigenTrust algorithm for reputation management is used to decrease the number of downloads of inauthentic files in a peer-to-peer file-sharing network [91]. PeerTrust is another research related to peer-to-peer networks that is supporting reputation-based trust for peer-to-peer electronic communities [92]. Developing trust computing mechanism based on risk evaluation properties is another trust research related to peer-to-peer networks [93]. Another model related to trust and risk in peer-to-peer networks for open environments is presented in [94]. Other trust models related to trust in networks is proposed in [95, 96]. Trust and risk are related concepts and there are misunderstanding in computer science related to the concepts. The conceptualization of trust, risk and their relationship in electronic commerce is studied in [97]. Other work related to trust and risk in computer science is in [98–102].

Trust is hot research topic for all areas of computer science. For instance, single-facet approach cannot capture the wide and varied range of subjective views of trust [103]. Therefore, a multi-faceted model of trust is proposed to cope with this case [103]. A conceptual software development process is presented by considering a client's perspective together with pointers for more general applications of the findings related to control, trust, and bargaining power in customized information system development [104]. In this study, the client was trying to find the optimal kind and level of trust and control in order to safeguard itself against any possible opportunism of the vendor. A social mechanism of reputation management based on trust is proposed to avoid undesired interactions between participants in [105]. In this research, people are represented with autonomous agents in electronic communities where agents interact. Experimental evaluation of trust is studied to show the validity of trust theories [106].

2.8. Some Open Research Problems Related to Trust

As trust is defined in a different manner and it is a matter in different fields of science, there are vast numbers of open research problems. However for the sake of brevity, we will mention only some of them in this thesis.

Recently, online systems have gained more attention from researchers who take trust into account as a research topic in computer science. For example, Massa explains some of the problems in online systems related to modeling and usage of trust in three groups [1]. The first group is concerned with the differences in how trust relationships are modeled in real and virtual world. It is natural that trust relationships in real world and in virtual world are different, but scientists want to represent trust relationships in virtual world to make them sensible for humans by using modeling techniques. The most relevant issues with this challenge are disproportion in positive trust, modeling negative trust, rigidity of language for expressing trust statement, single trust context, keywords for trust statements conveying undesired meaning, and etc. If the trust relationship information is available, exploiting trust is the next challenge that constitutes the second group of challenges [1]. Some problems in this group is creating scalable trust metrics and exploring negative trust statements. The last group of challenges

is related to identities, privacy, and attacks, which group contains enormous problems for modeling trust in computer based systems. Fake and multiple identities, attacks, portability and interoperability are some of the open problems in this group. Actually, some problems of this group may gain more attention than other problems about modeling trust such as vulnerability to attack.

Sun *et al.* define some attacks and their possible defense mechanisms for trust establishment systems in distributed networks in [18]. For instance, bad mouthing attack is described as *"dishonest recommendations that are provided by malicious parties to accuse good parties and boost trust values of malicious peers"*. The defense mechanism to this attack proposed in [18] is based on formally building and utilizing recommendation trust. Furthermore, the joint effects of various attacks can be an interesting future research topic.

There are other researches that address trust problems in online systems [4, 46, 50, 60, 107]. Josang *et al.* explain some problems and solutions in online systems that use trust management based on reputation systems [46]. Low incentive for providing rating is a problem where transaction partners do not have direct incentive for providing rating about the other party. The reason is that raters do not have any profit directly from providing ratings. Another problem is the bias towards positive ratings. In e-commerce systems, it is observed that only 0.6% of all ratings provided by buyers and only 1.6% of all the rating provided by sellers were negative [46]. Bias toward positive rating is explained as a hope of getting positive return, whereas fear of relation from the other party avoids negative ratings [46]. Some other problems that are explained are unfair ratings, change of identities, quality variations over time, discrimination, and so on [46]. The paradoxical effect of negative trust is emphasized and a method for trust network analysis is proposed using subjective logic [60].

Dynamic properties of trust relationships give rise to many problems. Changing values of weights assigned to information sources affect the level of reliability in networks security systems [58]. For instance, in ephemeral networks trust relationships have to be established and re-established frequently based on networks and perceived

environment. This means that they take into account dynamic factors such as time and location to calculate trust relations [39]. Additionally, Krukow also touches on the importance of dynamic properties of trust relationships on security policies, where he states that trust relationships change over time and they require continuous monitoring and re-evaluation [5]. Analyzing dynamic factors of trust relationships and exploring their mathematical properties are further open problems [18]. To handle dynamic properties of trust, such as history dependence property, learning and reasoning mechanisms must be used [19].

In literature, there are lots of other trust related problems, where they are generally defined within specific application domain. For example, heterogeneous trust is described as an open issue to implement data privacy policies [108]. Some other application specific problems are defined in [5, 7, 8, 19, 40, 49, 53, 54, 56, 59, 109].

3. TRUST CORE MODEL: A MODEL FOR TRUST ASSESSMENT OF SECURITY SYSTEM FROM ENTITY POINT OF VIEW

In this chapter, we propose the core model for assessing the trust of the security system of a service from an entity point of view. The model contains architecture of the trust assessment system of an entity. Novel trust assessment metrics are also introduced. The architectural approach is applicable on entities in open environments. The proposed architectural approach has been applied to Hospital Online Appointment Service as a case study with three scenarios. The case study has been evaluated experimentally with simulations. The experimental evaluation has shown that the proposed approach ensures trust assessments in open environments, where each entity may have different needs from the security system of a service. Moreover, the entity may assess the trust even with incomplete security information.

The rest of this chapter is organized as follows. We present our motivation and contributions in Section 3.1. We show some related work in Section 3.2. We present the architectural approach in Section 3.3. Section 3.4 describes the formal model of trust assessment environment. In Section 3.5, we present the formal representation of the security system of a service for trust assessments. Section 3.6 is about information sources. Section 3.7 and Section 3.8 describe our trust assessment metrics and the trust assessment process in sequence. We provide a complexity analysis of trust assessment in Section 3.9. In Section 3.10, we present Hospital Online Appointment Service as a case study with experimental evaluation. Finally, we conclude in Section 3.11.

3.1. Introduction

Emerging open environments are expected to be service-oriented environments that contain diverse number of services and autonomous entities. The web is an instance of service-oriented environment. World Wide Web Consortium (W3C) defines a

web service as a software system designed to support interoperable machine to machine interaction over a network. A web service is a set of related functionalities that can be manipulated over the web [110].

Since, the diversity of services increases in service-oriented environments, trust problems related to security issues become apparent. Computer security is discussed in the categories of authentication, confidentiality, integrity, and availability [2, 18]. On the other hand, trust has been investigated in various fields of science, such as philosophy and computer science [1, 57], but there is no consensus about trust issues in different fields of science even in the same field of science.

3.1.1. Motivation

The trustworthiness of a service mostly depends on its security system. A security system is a set of security mechanisms that are implemented according to a security policy. A security policy can be described as a collection of rules that allow or disallow possible actions, events, or something related to security about a service [2,3,7,8]. On the other hand, a security mechanism implements security policies in the system.

An entity needs to trust to security systems of services to interact with them according to its own needs in open environments. Then, the entity can make decisions for future interactions with the services. Particularly, the entity should assess the trust of the security system of a service before making interactions with the service. Therefore, assessing the trust of a security system from entity point of view is a challenging issue. The trust assessment necessitates precise trust computation models and specific trust assessment architectures of entities.

There are many trust computation models [1, 19, 46] that can be applied for assessing the security system of a service. However, existing trust computation models do not provide a solution to assess the trust of the security system from an entity point of view according to trust assessment architecture of an entity. Our motivation is the lack of a model and trust assessment architecture of an entity for assessing the trust

of the security system of a service from the entity point of view in open environments.

3.1.2. Contributions

In this chapter, we propose a model for assessing the trust of the security system of a service from an entity point of view. The model contains architecture for assessing the trust of the security system in an open environment. The architecture can be applied in any entity for trust assessments. An entity may assess the trust of all properties of the security system or some properties of the security system. The trust assessments are carried out according to our novel trust assessment metrics. We support the proposed model with a case study, where the behaviors of proposed metrics are analyzed via three scenarios and simulations. We can summarize contributions of our work as follows.

- We introduce a new architecture for assessing the trust of the security system of a service according to needs of an entity in open environments. The architecture provides a systematic way for trust assessment in entities.

- We propose two types of trust assessment metrics namely, partial metrics and total metrics. Partial metrics are used to assess a specific property of the security system of a service, whereas total metrics are used to assess all properties of the security system according to needs of a specific entity. The entity may use only those security properties to which it has trust for future interactions. Therefore, the proposed metrics can increase the possibility of interactions among entities and service in open environments.

- We present a step by step trust assessment process in a trust assessment system. The process shows the applicability of our model.

3.2. Related Work

Trust has been investigated in different application contexts of computer science and there are many trust based solutions in each application context. For example,

all components of network services are referred as a trust management problem, where there is an application to solve the trust management problem [54]. The trust issues in service-oriented environments are formally investigated in [55]. A methodology for revising and managing theories of trust for agent based systems is proposed in [111], where the methodology is concentrated to trust changes. Ad-hoc networks are another popular context for the trust research. For instance, trust metrics related to ad-hoc networks are evaluated in [112]. A more recent example related to trust based solution for ad-hoc networks is in [113], which is about trusted routing in mobile ad-hoc networks.

Although these researches contain many contributions to trust and trust related issues, none of them presents a solution for assessing the trust of a security system from an entity point of view.

3.3. The Architectural Approach for Trust Assessment

In our approach, trust assessments are carried out about the security system of a service according to needs of an entity. Therefore, our approach contains three main components, namely an entity, a service and a trust assessment system.

3.3.1. Place of Trust Assessment System

The trust assessment system can be either an independent system or a part of an entity or a service. Specifically, the trust assessment system can be a separate system independent from entities and services as shown in Figure 3.1a. In this case, the trust assessment system has to gather information for trust assessments both from entities and services by communicating with them. An entity has to submit its needs from the security system of a specific service to a trust assessment system. Moreover, the entity has to inform the trust assessment system about its utility relations with services for computations of trust metrics. Actually, the utility relation between an entity and a service is a private relation of the entity so that entity may not want to reveal its utility relation with the service. On the other hand, the trust assessment

system has to obtain information about security systems of services from the services. Both entities and services have to communicate with the trust assessment system in this case, where the communication security may be an important issue. In addition, the trust assessment system has to be a trustworthy system for entities and services, which is usually impossible for open environments.

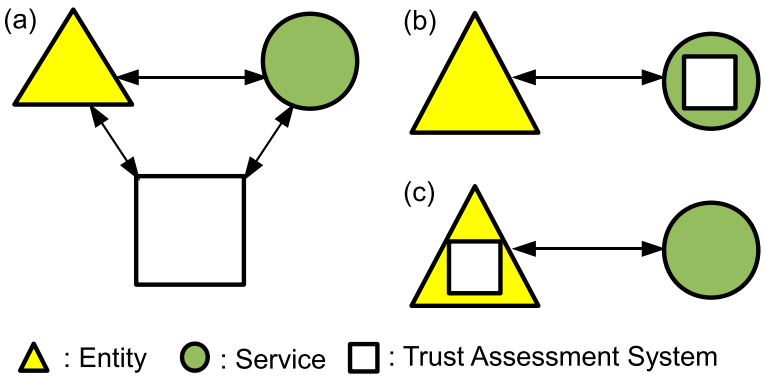

Figure 3.1. The location of a trust assessment system (a) independent from entities and services (b) in a service (c) in an entity.

Services may assess their own trust according to their security systems by considering needs of entities from their security systems. In this case, each service has to have a trust assessment system as shown in Figure 3.1b. Entities have to inform services about their needs from the security systems of services. Moreover, each service has to know the computation method of trust metrics specific to an entity. Similar to the independent trust assessment system case as shown in Figure 3.1a, communication security and privacy problems may occur in this case.

In the last case, each entity has a trust assessment system as shown in Figure 3.1c. An entity assesses the trust of the security system of a service according to its own needs by itself. Therefore, the entity does not need to send its private information to another party for trust assessments, where the privacy of an entity related to computations of trust assessment metrics are preserved. Because an entity does not need to send information to another party, communication security problems decrease. For these reasons, our trust assessment system is in an entity and here we present the architecture of the trust assessment system of an entity.

Each trust assessment model needs information for computations of trust metrics. Therefore, information gathering is an important issue for any trust computation model. Additionally, the amount of information is also significant for trust computations. In our approach, an entity has to gather information for trust assessments. We presented a model for security information flow on entities for trust computations in [26]. Moreover, an entity obtains security information from services and entities according to the security information flow model presented in [26].

3.3.2. The Architecture

The architecture of Trust Assessment System of an entity for assessing the trust of the security system of a service consists of four layers, namely Communication Interface Layer, Information Classification Layer, Trust Assessment Layer and Decisions Database Layer. Trust Assessment System Architecture is shown in Figure 3.2, which is an improved form of the architecture shown in [23].

Communication Interface Layer is responsible to manage interactions with entities and services. It interacts with other entities and services to obtain information for trust assessments. Obtained information is submitted to Information Classification Layer. Communication Interface Layer gets trust assessment results from Decisions Database Layer and returns them to the entity after each trust assessment.

Information Classification Layer classifies existing information in an entity. Since the trust assessment process is time dependent, information for future trust assessments has to be extracted and classified to carry out accurate trust assessments. Existing information originates from other entities and services. Moreover, the entity generates information related to security systems of services by its own observations according to its private needs. Information Classification Layer also manages old trust assessment decisions for subsequent computations of trust metrics. Briefly, Information Classification Layer determines values of all parameters that are used for computations of trust metrics. Additionally, if the values of the parameters are unknown, they are set to their default values.

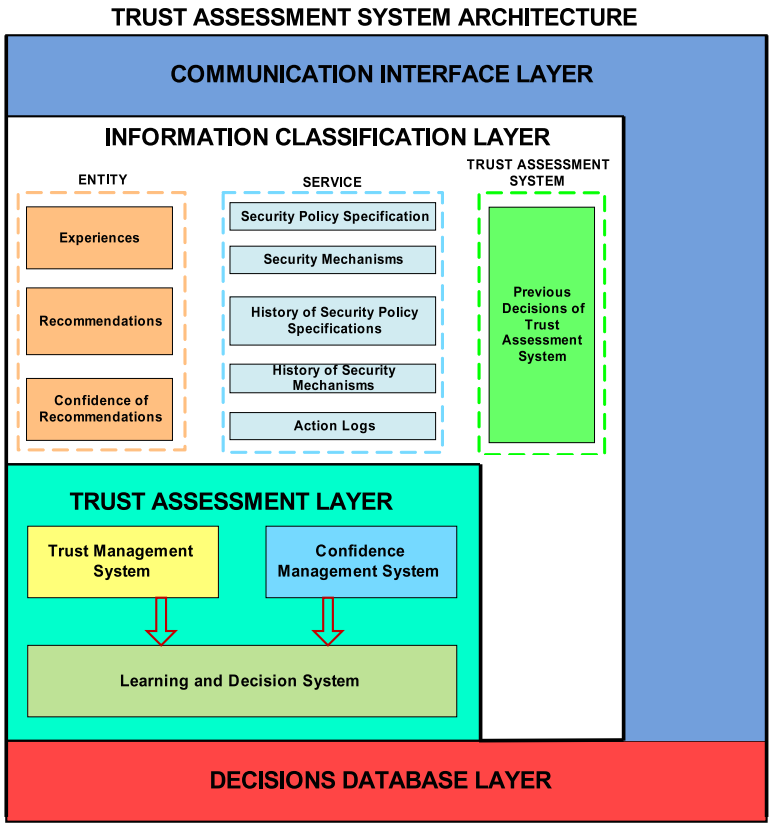

Figure 3.2. The architecture of the trust assessment system in an entity.

Trust Assessment Layer contains three systems to compute trust metrics. Computations of trust metrics are carried out in Trust Assessment Layer. *Trust Management System* computes Partial Trust Level and Total Trust Level metrics. *Confidence Management System* is responsible to compute confidences of metrics that are computed in Trust Management System. There are two confidence metrics, namely Partial Confidence metric and Total Confidence metric. *Learning and Decision System* computes relative trust assessment metrics according to assessed trust metrics and their confidences. Additionally, old relative assessments contribute to computations of relative trust metrics. Partial Relative Trust Decision and Total Relative Trust Decision are metrics computed in Learning and Decision System.

The bottom layer in our architecture is *Decisions Database Layer* that is responsible to store all old assessments. Decisions Database Layer assists Information Classification Layer to extract old assessment information for next computations of metrics. Simply, Decisions Database Layer can be classified as the database of Trust

Assessment System of an entity.

3.4. Formal Representation of Trust Modeling Environment

In early days, the amount of digital communication was limited to large institutions only and the cost of the digital communication was very high. However, with the rapid development of computer networks the cost of digital communication has decreased dramatically. Therefore, computer networks have become more common than ever. For example, the increasing availability of Internet connections with lower cost allows regular computer users to easily communicate with each other by using their personal computing platforms, such as desktop computers, mobile computers, smart phones, and etc.

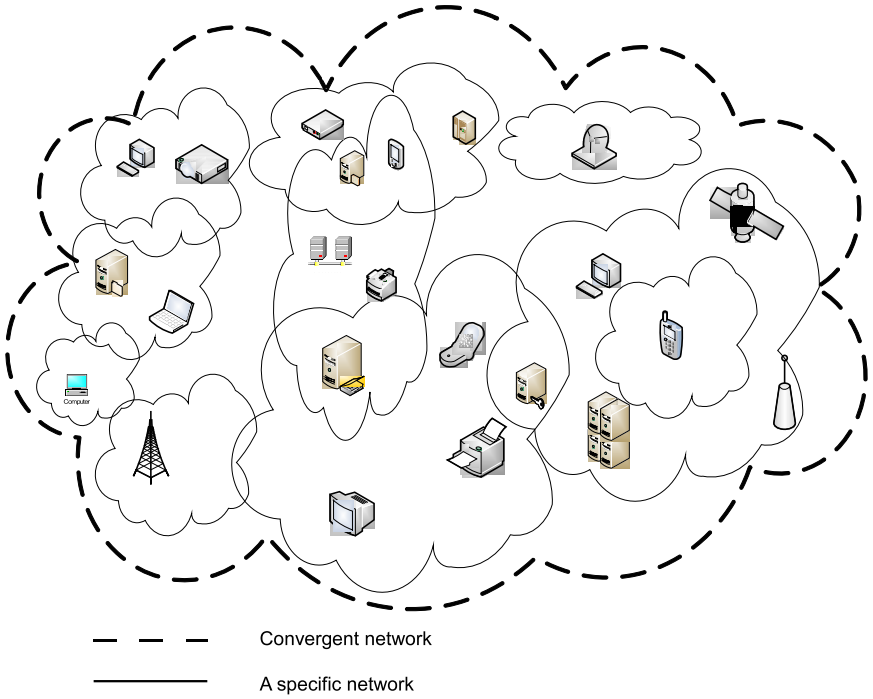

Figure 3.3. A convergent network containing different devices on which services may run.

Current developments in computer networks enable one person to connect many networks. Moreover, a person may use a single device to obtain services from different networks. The networks should be interconnected to ensure the access to services. Moreover, most of services should be accessible without considering the underlying

network. For example, a web service may be accessible with a cell phone, which has a Wi-Fi interface, by using either a 3G network or a Wi-Fi network.

Emerging networks are expected to interwork with each other and converge to a ubiquitous network, where the network is an open environment. Moreover, convergent networks may contain various devices as shown in Figure 3.3. The networks have to support diverse number of services to meet demands of users. For example, a person may use a desktop computer in a local area network to access the Internet and buy an airline ticket or the person may use a cell phone to connect to a 3G network to carry out such operation. Both the desktop computer and the cell phone have to access the web service for online tickets. Briefly, emerging open environments are expected to be service-oriented environments that contain diverse number of services and autonomous entities [114]. The web is an instance of service-oriented environment.

In our model, the environment is a service-oriented environment. The service-oriented environment consists of two types of nodes, entities and services. Entities can be autonomous software agents. The software agents may represent people or business enterprises, whereas services are responsible to provide services to entities. Actually, software agents are able to receive and provide services. However, we assume that entities can get services only and entities cannot provide any service.

Our goal in this chapter is to formally define the environment that represents emerging convergent networks as an overlay network. To accomplish this goal, we formally define the convergent network as an overlay network for the computation of trust of a security system. Additionally, we define components of the overlay network in a formal manner.

The following section formally defines an entity that computes trust of the security system of a service. Next section formally defines a service. The third section covers formal definitions of interactions between a service and an entity and formal definitions of interactions among entities. Then, we define the overlay network in subsequent section. Finally, we conclude the chapter.

3.4.1. Entity

Entities are autonomous agents or software applications representing users. For instance, an entity may be a web application interface to the Internet. The application is used by a person to get a service. An entity may also be a software agent that is responsible to accomplish a given task autonomously.

An entity may have many properties however we only consider security properties of the entity in this model. Therefore, other properties of an entity, such as communication interface of the entity, are out of scope of this thesis.

An entity has security evaluation information from different sources, where the entity obtains security evaluation information directly by observations and indirectly from other entities [26]. Each service can also send its own security evaluation information to the entity.

An entity interacts with many services and observes behaviors of security systems of the services. The entity has direct security evaluation information for trust computations that we call experiences if the entity has previously interacted with services.

Entities can receive security evaluation information about security systems of services from other entities. We call recommendations for such information received from entities. A recommendation is a part of indirect information, which is used for trust computations. An entity may receive recommendations from some entities and may send its recommendations to other entities. The entity sends recommendations to other entities according to utility relations with these entities.

An entity may interact with a wide variety and a large number of entities from which the entity receives recommendations. The multiplicity of entities may cause confidence problems about received recommendations. The confidence of a recommendation is the quality measure of the recommendation that an entity sends to other entities. An entity sends recommendations and the confidences of recommendations

to other entities. A confidence of recommendation is a part of indirect information received by an entity.

In this thesis, an entity is formally represented with α in a service-oriented environment. The set of entities are represented with $A = \{\alpha_1, \ldots, \alpha_n\}$ in the environment, where $n \in \mathbb{Z}^+$. Set A is normally very large. Therefore, an entity is expected to interact with some entities in the environment, which is a subset of set A. Moreover, each entity may interact with different entities.

3.4.2. Service

A service provides different duties to entities. In this thesis, we are interested in security aspects of a service. The security system of an online ticked reservation service is an instance of a service. The security system of a service contains a security policy, security mechanisms and action logs. We assume that the security policy is well defined by a security policy specification.

A security policy specification shows the expected behavior of the security system of a service under different circumstances. For example, assume that a service has a password based access control security mechanism. Additionally, the security policy specification of the service has a rule that necessitates the length of a password to be at least x characters. If an entity attempts to set a password that is less than x characters, the security mechanism has to reject the password setting.

The security system of a service contains many security mechanisms. For instance, a service may have many cryptographic algorithms for encryption of sensitive data, where each algorithm is a security mechanism. Moreover, the encryption keys may be stored in a hardware module, such as in a Trusted Platform Module (TPM). A TPM is also a security mechanism in our approach.

Each service monitors actions of its security system so entities generate actions logs according to the monitoring process. The action logs in a service consist of infor-

mation related to the security system of the service.

The security system of a service is dynamic in our approach. Therefore, the security policy of a service may change with time. Moreover, security mechanisms of a service may vary in time.

We represent a service with ω in a formal manner in service-oriented environment. We also represent the set of services with $\Omega = \{\omega_1, \ldots, \omega_m\}$ in the environment, where $m \in \mathbb{Z}^+$. Set Ω, which may contain many applications, such as the security system of a web service or the security system of an application in a mobile device, is normally very large. However, an entity consider only the services it interacts to compute the trust of the security system of a specific service.

3.4.3. Interactions

An entity obtains security evaluation information with interactions. We have two types of interactions from the entity point of view, namely entity-service interaction and entity-entity interaction. An entity can interact with any service in service-oriented environment. The entity can also interact with any other entity in the environment. Set $R = \{r_1, \ldots, r_z\}$ represents the set of interactions between entities and services and interactions between two entities, where $R \subseteq \{A \times \Omega\} \cup \{A \times A\}$ and $z \in \mathbb{Z}^+$.

An entity can interact with many services and can obtain direct security evaluation information from the services. The entity evaluates the obtained security evaluation information according to its interest relationship with each service separately. For example, assume that entity α has security evaluation information about security systems of services ω_k, ω_b, and ω_c. The goal of entity α is to gather as much as possible security evaluation information about service ω_k. Therefore, each security evaluation information obtained from other services contributes differently to the computation of direct security evaluation information about service ω_k. Formally, interaction r exists between entity α and service ω if entity α obtains direct information from service ω at a specific time, where $\alpha \in A$, $\omega \in \Omega$, and $r \in R$.

Entities can interact with each other in a service-oriented environment and they can obtain indirect security evaluation information. Similar to entity-service interaction, the entity evaluates information obtained from each entity according to its interest relationship with each entity separately. An entity-entity interaction r exists between entity α_i and entity α_j according to the point of view of entity α_i if entity α_i obtains indirect information related to services from entity α_j at a specific time, where $\alpha_i, \alpha_j \in A$ and $r \in R$.

3.4.4. Formal Representation of Overlay Network

The environment is an overlay network containing services and entities over a convergent network as in Figure 3.4. We call entity-service overlay networks for such overlay networks. Additionally, we use overlay network and entity-service overlay network interchangeably in this thesis. The formal definition of an entity-service overlay network is as follows.

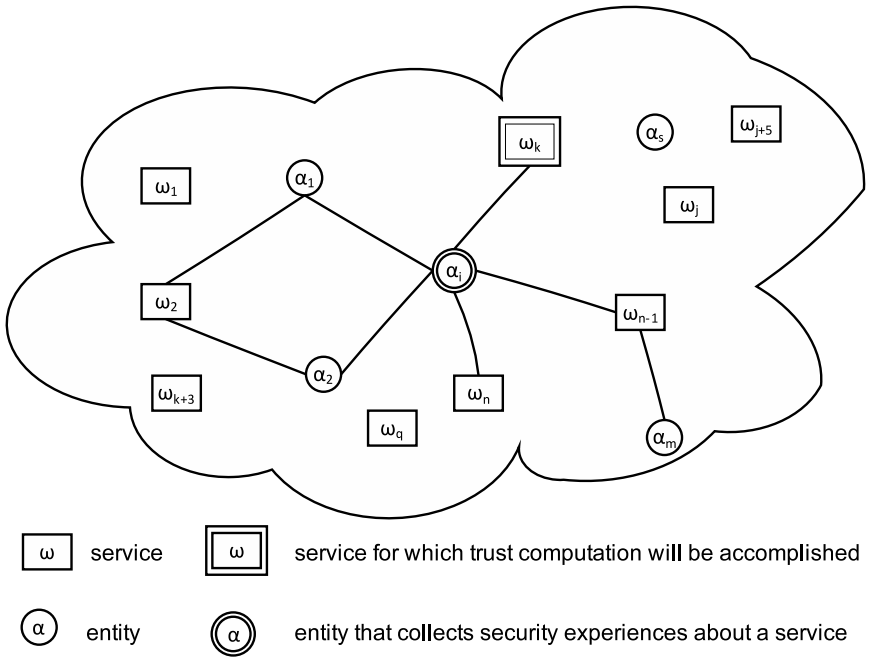

Figure 3.4. An entity-service overlay network for obtaining security evaluation information from different services.

Definition 3.1 *$A = \{\alpha_1, \ldots, \alpha_n\}$ represents the set of entities, $\Omega = \{\omega_1, \ldots, \omega_m\}$ represents the set of services, and $R = \{r_1, \ldots, r_z\}$ represents the set of interactions be-*

tween entities and services, where $R \subseteq A \times \Omega$ *and* $n, m, z \in \mathbb{Z}^+$. *Then, an entity-service overlay network* N *for obtaining security experiences from services for computations of trust is tuple* $N = (A, \Omega, R)$.

Set Ω, which may contain many applications, such as the security system of a web service or the security system of an application in a mobile device, is normally very large. Similarly, there is large number of entities. However, we consider only one entity and the services the entity interacts to compute trust of the security system of a specific service. In addition, set R contains only security related interactions between entities and services.

3.5. Formal Representation of a Security System for Trust Modeling

Services send information about their security systems to entities. Additionally, each entity observes behaviors of services and generates information about security systems of the services. An entity also receives information about security systems of services from other entities.

We represent the security system of a service from the entity point of view with atomic units. Therefore, entities represent the security system of a service based on their own needs. Each entity generates information about all atomic units of a service by observations and by receiving information from other entities for trust assessments as in [26]. An entity can observe only security mechanisms of a service as the security system of the service. Therefore, the atomic unit representation of the security system of a service is the representation of security mechanisms with atomic units.

A set of atomic units in an entity is represented with $\Phi(t) = \{\varphi_1, \ldots, \varphi_n\}$, where $n \in \mathbb{Z}^+$. φ_i denotes an atomic unit of set $\Phi(t)$. Security mechanisms of services are dynamic therefore the set representation of a security system depends on time. In our set model, t denotes the time varityping purpose of a security mechanism from the entity point of view. For example, assume that the security system of an airline

reservation web service uses an encryption mechanism to store its internal data. The service may use DES symmetric key algorithm, AES128 or AES256 asymmetric key algorithms for encryption. In addition, the service may use one of two different methods for authentication, which are the digital signature based authentication method DSS or the password based authentication method PW. In this case, set $\Phi_{air}(t)$ has five atomic units, $\Phi_{air}(t) = \{DES, AES128, AES256, DSS, PW\}$.

Each entity may have different granularity for the representation of atomic units due to resource costs of entities. It is expected that if all entities have different granularity of atomic units according to their needs, the atomic unit representation of a security system will lead to better computations of trust. For instance, symmetric encryptions may perform better than asymmetric ones for a resource limited entity, therefore each encryption method has to be better represented with an atomic unit. On the other hand, another entity may not have a resource limitation so that the entity may use any encryption method and all encryption methods may be represented with one atomic unit, such as $\Phi_{air}(t) = \{ENC, DSS, PW\}$, where in this case, the atomic unit ENC represents DES, AES128, and AES256 in an aggregated form.

3.6. Information Sources

There are three sources of information for computations of the metrics. Information obtained by observations and information received from other entities is the first information source. Such information depends on the subjective perception of an entity, which we call *perceived information*. The second source is services that send information about their security systems directly to an entity. The last information source is the TAS of an entity, where old assessments are stored. Each metric can be computed according to the three information sources.

We represent the perceived information related to atomic unit φ_y at a given time t with $\iota^{\alpha}_{x,y}(t) \in [0,1]$, where $\varphi_y \in \Phi_x(t)$. In addition, we represent the perceived information related to all atomic units of service ω_x with $\iota^{\alpha}_{x}(t) \in [0,1]$. For instance, the ultimate experience is a kind of perceived information. We present the computation

of ultimate experience in Chapter 5.

Information received from services contributes to the computation of the metrics. $\iota^{\omega}_{x,y}(t) \in [0,1]$ represents information that is computed with information received from service ω_x about atomic unit φ_y at a given time t. An entity computes information $\iota^{\omega}_{x}(t) \in [0,1]$ about all atomic units of service ω_x. For example, extracted trust information from the security system of a service is such kind of information. The computation of the extracted trust information is presented in Chapter 4.

New values of the trust metrics depend on their former values. The effects of the former values to computations of new values related to atomic unit φ_y is represented with $\iota^{\chi}_{x,y}(t) \in [0,1]$, where $\varphi_y \in \Phi_x(t)$. Additionally, former values of trust metrics affect computations of total metrics. We represent such effect with $\iota^{\chi}_{x}(t) \in [0,1]$ about service ω_x.

Values of information metrics represent the usefulness of information for computations of the trust metrics. Particularly, if a metric approaches to one or it is one, the information has maximum usefulness for trust computations. On the other hand, low values mean that there is a lack of information or the information may not be useful for trust computations. The meanings of values related to specific security information metrics are explained in more details in Chapter 4 and Chapter 5, such as the value of ultimate experience.

3.7. Trust Assessment Metrics

In our trust assessment approach, an entity can assess both trust of a specific security property and trust of all security properties. We define two types of trust assessment metrics, namely partial metrics and total metrics. Partial metrics are about a specific security property whereas total metrics are about all security properties of a service.

Specifically, we define six trust metrics to assess trust of the security system

of a service in an entity. In relation to partial and total metrics, we define trust level metrics, confidence metrics and relative trust assessment metrics. Each metric is computed according to needs of an entity so that values of metrics are specific to the entity. Partial metrics are about a specific atomic unit of a security system and they constitute half of trust metrics. On the other hand, total metrics are about all atomic units of a security system.

3.7.1. Trust Level

Trust level metrics are computed in Trust Management System. Partial Trust Level of atomic unit φ_y is represented with $\gamma_{x,y}^{PT}(t) \in [0,1]$ at a given time t and is computed as follows, where $\varphi_y \in \Phi_x(t)$.

$$\gamma_{x,y}^{PT}(t) = co_{x,y}^{\alpha} \iota_{x,y}^{\alpha}(t) + co_{x,y}^{\omega} \iota_{x,y}^{\omega}(t) + co_{x,y}^{\chi} \iota_{x,y}^{\chi}(t). \tag{3.1}$$

Partial Trust Level metric shows the trust level assessed by a specific entity about a particular atomic unit of the security system of a service by considering all available information related to the atomic unit without taking into account the confidence of the assessment. The coefficients $co_{x,y}^{\alpha}, co_{x,y}^{\omega}, co_{x,y}^{\chi} \in [0,1]$ show the impacts of information sources for the computation of $\gamma_{x,y}^{PT}(t)$, where $co_{x,y}^{\alpha} + co_{x,y}^{\omega} + co_{x,y}^{\chi} = 1$.

On the other hand, an entity uses Total Trust Level metric to assess the trust of all atomic units of the security system of service ω_x at a given time t. Total Trust Level about the security system of service ω_x is represented with $\gamma_x^{TT}(t) \in [0,1]$ and is computed as follows.

$$\gamma_x^{TT}(t) = co_x^{\alpha} \iota_x^{\alpha}(t) + co_x^{\omega} \iota_x^{\omega}(t) + co_x^{\chi} \iota_x^{\chi}(t). \tag{3.2}$$

Total Trust Level is a metric that shows the trustworthiness of the security system of a service according to needs of a specific entity without taking into account the confidence of the assessment. In contrast to Partial Trust Level metric, Total Trust Level metric is computed by considering all information about the security system of a service. Therefore, Total Trust Level metric is significant for an entity if the entity needs to assess the trustworthiness of the whole security system of a service. Similar to the coefficients of $\gamma_{x,y}^{PT}(t)$, the coefficients $co_x^{\alpha}, co_x^{\omega}, co_x^{\chi} \in [0,1]$ show the impacts of information sources for the computation of $\gamma_x^{TT}(t)$, where $co_x^{\alpha} + co_x^{\omega} + co_x^{\chi} = 1$.

Values of trust level metrics show the amount of trustworthiness. If the value of a metric is high, the security system or the atomic unit is more trustworthy. For instance, if $\gamma_x^{TT}(t) = 1$, the security system of service ω_x has maximum trust. Whereas, if $\gamma_x^{TT}(t) = 0$, the security system of service ω_x has minimum trust or no trust.

3.7.2. Confidence

An entity computes Partial Trust Level and Total Trust Level without considering the quality of these computations. In our approach, Partial Confidence and Total Confidence metrics are quality metrics of Partial Trust Level metric and Total Trust Level metric respectively. Specifically, a confidence metric shows the quality of the computation of a trust level metric at a specific time. In the proposed architecture, confidence metrics are computed in Confidence Management System.

Partial Confidence metric is the quality measure of Partial Trust Level metric $\gamma_{x,y}^{PT}(t)$ at a given time t and is represented with $\gamma_{x,y}^{PC}(t) \in [0,1]$. In our approach, if perceived information of an atomic unit, information received from a service related to the atomic unit, and information extracted from former assessments related to the atomic unit differ considerably, the confidence to Partial Trust Level of the atomic unit is expected to be low. Perceived information, information received from a service and information extracted from previous assessments lead to our new concept *Information Difference*. *Information Difference* is significant for the computation of Partial Confidence metric. The information difference $iDif_{x,y}(t)$ related to atomic unit φ_y of service

ω_x is computed as in Equation 3.3.

$$iDif_{x,y}(t) = \left|\iota^{\alpha}_{x,y}(t) - \iota^{\omega}_{x,y}(t)\right| + \left|\iota^{\alpha}_{x,y}(t) - \iota^{\chi}_{x,y}(t)\right| + \left|\iota^{\omega}_{x,y}(t) - \iota^{\chi}_{x,y}(t)\right|. \tag{3.3}$$

Partial Confidence $\gamma^{PC}_{x,y}(t)$ about atomic unit φ_y of service ω_x is computed as follows.

$$\gamma^{PC}_{x,y}(t) = \begin{cases} 0 & , iDif_{x,y}(t) > 1 \\ 1 - iDif_{x,y}(t) & , otherwise \end{cases} \tag{3.4}$$

Total Confidence metric is the quality measure of Total Trust Level metric $\gamma^{TT}_{x}(t)$ at a given time t and is represented with $\gamma^{TC}_{x}(t) \in [0,1]$. Total Confidence is computed as in Equation 3.6. $iDif_x(t)$ represents information difference related to the security system of service ω_x that is computed follows.

$$iDif_x(t) = |\iota^{\alpha}_{x}(t) - \iota^{\omega}_{x}(t)| + |\iota^{\alpha}_{x}(t) - \iota^{\chi}_{x}(t)| + |\iota^{\omega}_{x}(t) - \iota^{\chi}_{x}(t)|. \tag{3.5}$$

$$\gamma^{TC}_{x}(t) = \begin{cases} 0 & , iDif_x(t) > 1 \\ 1 - iDif_x(t) & , otherwise \end{cases} \tag{3.6}$$

The computation of Total Confidence metric is the same as the computation of

Partial Confidence metric. If Total Trust Level metric is computed with high confidence, Total Confidence metric related to Total Trust Level metric is high. For instance, if $\gamma_x^{TC}(t) = 1$, $\gamma_x^{TT}(t)$ has maximum confidence whereas if $\gamma_x^{TC}(t) = 0$, $\gamma_x^{TT}(t)$ has no confidence.

Entities are optimistic in our approach in the sense that an entity should relay on its initial computations of trust if it has no previous information. Therefore, if an entity does not have both perceived information and information received from a specific service and if the entity does not have trust assessments information related to the security system of the service, the entity will have maximum confidence to the newly computed trust level metrics.

3.7.3. Relative Trust Assessment

An entity may need to consider both a trust level and its confidence to make a decision. Trust level metrics are combined with related confidence metrics and the combination is represented with relative trust assessment metrics.

Similar to trust level metrics and confidence metrics, we have two relative trust assessment metrics, namely Partial Relative Trust Assessment metric and Total Relative Trust Assessment metric. *Partial Relative Trust Assessment* metric represents the assessed trust of an atomic unit of a service in an entity at a given time t. The metric contains the confidence of the assessment. Partial Relative Trust Assessment metric is represented with $\gamma_{x,y}^{PR}(t) \in [0,1]$ and is computed as follows, where $\varphi_y \in \Phi_x(t)$.

$$\gamma_{x,y}^{PR}(t) = \kappa_{x,y}\gamma_{x,y}^{PT}(t)\,\gamma_{x,y}^{PC}(t) + (1-\kappa_{x,y})\,\Psi_{x,y}(t)\,. \tag{3.7}$$

$\gamma_{x,y}^{PR}(t)$ is computed according to $\gamma_{x,y}^{PT}(t)$ and $\gamma_{x,y}^{PC}(t)$. Former values of $\gamma_{x,y}^{PR}(t)$ also affect next computations of $\gamma_{x,y}^{PR}(t)$. $\Psi_{x,y}(t) \in [0,1]$ represents the effect of former

assessments about $\varphi_y \in \Phi_x(t)$ as in Equation 3.7. Each entity may compute $\Psi_{x,y}(t)$ differently. Each entity may also evaluate recently computed information and former assessments differently for the computation of $\gamma_{x,y}^{PR}(t)$. The coefficient $\kappa_{x,y} \in [0,1]$ represents the significance of recent information for the computation of $\gamma_{x,y}^{PR}(t)$.

Total Relative Trust Assessment metric represents the assessed trust of all atomic units of the security system of a service in an entity with the confidence of the assessment at a given time t. Total Relative Trust Assessment metric related to service ω_x is represented with $\gamma_x^{TR}(t) \in [0,1]$ and is computed as follows.

$$\gamma_x^{TR}(t) = \sum_{\forall \varphi_y \in \Phi_x(t)} \zeta_{x,y} \gamma_{x,y}^{PR}(t). \tag{3.8}$$

$\gamma_x^{TR}(t)$ metric is the weighted sum of all atomic units of a security system, where $\zeta_{x,y}$ is called *Impact Factor* representing the weight. Each atomic unit has different impacts for the computation of Total Relative Trust Assessment metric. Impact Factor $\zeta_{x,y} \in [0,1]$ about atomic unit φ_y is subjective and varies according to each entity, where $\sum_{\forall \varphi_y \in \Phi_x(t)} \zeta_{x,y} = 1$. If $\zeta_{x,y} = 0$, atomic unit φ_y is not considered for the computation of $\gamma_x^{TR}(t)$ in an entity. Otherwise, atomic unit φ_y contributes to the computation of $\gamma_x^{TR}(t)$ proportional to $\zeta_{x,y}$.

Values of relative trust assessment metrics determine the degree of trustworthiness. For example, $\gamma_{x,y}^{PR}(t) > \gamma_{x,z}^{PR}(t)$ means that atomic unit φ_y has more relative trust than atomic unit φ_z according to needs of an entity.

3.8. Trust Assessment Steps in an Entity

Each entity assesses the trust of the security system of a specific service according to its own needs from that service. Here, we present steps of trust assessments according

to our architectural approach to clarify the assessment process. The steps of the trust assessment process in Trust Assessment System of an entity, which is shown in Figure 3.2, are as follows.

Step 1: (Communication Interface Layer) TAS sends a message and then receives a message to/from a service and an entity about the security system of a specific service. TAS carries out this message exchange with all services and entities it interacts. Then, TAS checks the information completeness and freshness about the service. It determines incomplete information if exists. If there is insufficient information after the message exchange, TAS may repeat the message exchange several times.

Step 2: (Information Classification Layer) TAS classifies information obtained by observations related to the security system of the service. TAS also classifies the information received from other entities related to the service. Simply, TAS classifies the perceived information.

Step 3: (Information Classification Layer) TAS classifies information received from the service according to needs of the entity.

Step 4: (Information Classification Layer) TAS classifies previous assessment results related to the security system of the service according to present needs of the entity.

Step 5: (Information Classification Layer) If information for next assessment process is not complete, TAS uses default values for assessments.

Step 6: (Trust Assessment Layer) TAS computes both trust level metrics and confidence metrics related to the service by considering the present needs of the entity.

Step 7: (Trust Assessment Layer) TAS computes relative trust assessment metrics according to recent values of trust level metrics and confidence metrics, which are computed in previous step.

Step 8: (Decisions Database Layer) TAS stores recently computed values of all trust assessment metrics.

Step 9: (Communication Interface Layer) TAS returns values of newly computed trust assessment metrics to the entity.

The trust assessment steps given here are specific for Trust Assessment System that is in an entity. If Trust Assessment System is outside from an entity as shown in Figure 3.1a and Figure 3.1b, steps of the trust assessment process will be different.

3.9. Complexity of Trust Assessment

Trust assessment metrics are computed according to security evaluation information. The amount of security evaluation information determines the precision of a trust assessment. Therefore, obtaining security evaluation information is the major issue that determines the complexity of trust assessments.

In our approach, there are three kinds of information for assessing the trust of a security system of a service. Perceived information, $\iota^{\alpha}(t)$, is obtained from entities and services. The second kind of information, information received from services $\iota^{\omega}(t)$, is obtained only from services. The entity is the source of former trust assessments, $\iota^{\chi}(t)$. Therefore, obtaining all these three types of information determine the complexity of trust assessments.

An entity is interacting with other entities and services to obtain security evaluation information. The interaction is carried out by exchanging messages. The number of entities and the number of services with which the entity interacts increase the complexity of trust assessments. Let n_e represents the number of entities interacting with a specific entity and let n_s represents the number services interacting with the entity. Then, the complexity of a trust assessment is $O(n_e + n_s)$. Let $n = n_e + n_s$, then the complexity of a trust assessment is $O(n)$.

In our approach, each entity computes trust assessment metrics in a specific time period, for example each second. Moreover, an entity can exchange a number of messages to obtain security evaluation information both from entities and from services according to the trust assessment algorithm introduced in Section 3.8. Normally, an entity sends a message and receives a message, totally two messages per unit time, for each computation of trust metrics so the actual complexity of trust assessment is $O(2n)$. In exceptional cases, the entity repeats the message exchange several times. Let m represents maximum number of possible message exchange in an entity. Then, the complexity of trust assessment is $O(2mn)$.

The number of former trust assessments is also significant to determine the complexity of a trust assessment. Let n_{fc} be the maximum number of former trust assessments related to the security system of a specific service in an entity. Then, the complexity of trust assessment is $O(2mn + n_{fc})$.

Maximum number of messages m and maximum number of former trust assessments n_{fc} are constant. Therefore, precisions and complexities of trust assessments depend on n so $O(2mn + n_{fc}) = O(n)$. This result shows that the complexity of trust assessment increases linearly with n therefore our approach is scalable.

3.10. Case Study: Hospital Online Appointment Service

In this section, we analyze the proposed architectural approach based on information obtained from different sources according to our new trust metrics. The case study contains three scenarios, where an entity that represents a patient assesses the trust of the security system of Online Appointment Service of a hospital. In the first scenario, the entity uses perceived information and information extracted from old assessments for trust assessments. In the second scenario, the entity assesses the trust of the security system by using information received from the service and information extracted from old assessments. In the last scenario, the entity carries out trust assessments according to all existing information related to the security system of Online Appointment Service.

We simulate the scenarios and show the performance results of the proposed model. Simulations are carried out by using MATLAB R2009b version 7.9.0.529 that run on a PC with Intel Core 2 Duo E8400 3.00GHz processor and 3GB of RAM.

Assume that a patient has dental problems and wants to get an appointment from the dental clinic of a hospital. The patient is a busy person and can get an appointment only by using Online Appointment Service of a hospital. There are many hospitals with dental clinics from which the patient can get an appointment. However, the patient knows that some hospitals have weak security systems of their appointment services. A weak security system may result in privacy problems. Therefore, the patient needs to assess the trust of the security system before getting an appointment.

The patient has a software agent that represents herself. The software agent can assess the trust of the security system of a service according to privacy needs of the patient. Then, the patient can decide whether to get an appointment or not by considering the trust assessments related to the software agent. The software agent (SA) in this case study is the entity and Trust Assessment System architecture of the entity is implemented according to our architectural approach.

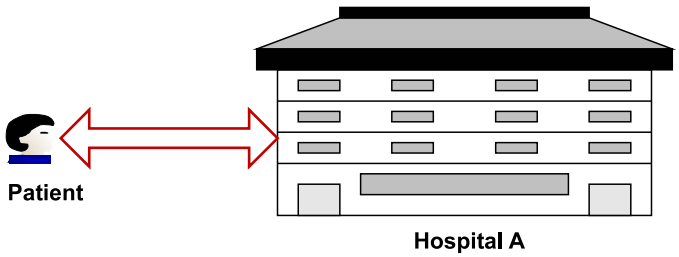

Figure 3.5. Getting an appointment from Online Appointment Service of Hospital A.

Assume that the entity carries out trust assessments about the security system of Online Appointment Service of Hospital A as shown in Figure 3.5. Trust assessments are carried out according to available information for every second. In this thesis, time is discrete and increases every second by one.

The entity represents the security system of Online Appointment Service of Hospital A with two atomic units $\{ENC, AC\}$. Set atomic units of the security system

are $\Phi_{HA}(t) = \{\varphi_1, \varphi_2\}$. Atomic unit ENC represents all encryption algorithms of the security system that is used to record personal data of a patient. Atomic unit AC represents the access control mechanism of the security system. For the sake of simplicity, we assume that the atomic unit representation of the security system does not change in the case study.

We are interested in the effects of perceived information and information received from services. Therefore, we select effects of former trust assessments to be constant. Specifically, we set $\iota^{\chi}_{HA}(t)$, $\iota^{\chi}_{HA,1}(t)$, and $\iota^{\chi}_{HA,2}(t)$ to be all 0.1. We assume that the effects of information sources are equal so coefficients $co^{\alpha}_{x,y}, co^{\omega}_{x,y}, co^{\chi}_{x,y}$ and $co^{\alpha}_{x}, co^{\omega}_{x}, co^{\chi}_{x}$ are all 1/3. We also assume that the significance of recent information is equal to the significance of old information so that $\kappa_{x,y}$ is 0.5. In addition, $\Psi_{HA,1}(t) = 0.05$ and $\Psi_{HA,2}(t) = 0.01$. Furthermore, $\zeta_{HA,1} = 0.35$ and $\zeta_{HA,2} = 0.65$.

3.10.1. Scenario 1: Assessments by Using Perceived Information and Information Extracted from Old Assessments

In this scenario, SA does not have information received from Online Appointment Service and it carries out trust assessments by using perceived information and information extracted from old assessments. Therefore, $\iota^{\omega}_{HA}(t)$, $\iota^{\omega}_{HA,1}(t)$, and $\iota^{\omega}_{HA,2}(t)$ are all zero. The aim of this scenario is to show the effects of perceived information.

Each entity can compute perceived information differently by considering its own needs so that $\iota^{\alpha}_{HA}(t)$, $\iota^{\alpha}_{HA,1}(t)$, and $\iota^{\alpha}_{HA,2}(t)$ depend on entities. In this scenario, we assume that $\iota^{\alpha}_{HA,1}(t)$ is between 0.3 and 0.4 and $\iota^{\alpha}_{HA,2}(t)$ is between 0.2 and 0.25. Only first ten values of $\iota^{\alpha}_{HA,1}(t)$ and $\iota^{\alpha}_{HA,2}(t)$ are outside of these boundaries, because the entity learns from perceived information. The first ten values of $\iota^{\alpha}_{HA,1}(t)$ are set to be 0.05, 0.09, 0.13, 0.16, 0.18, 0.20, 0.22, 0.27, 0.28, and 0.29 in sequence. And the first ten values of $\iota^{\alpha}_{HA,2}(t)$ are also set to be 0.01, 0.05, 0.06, 0.08, 0.09, 0.14, 0.18, 0.19, 0.199, and 0.1992. In this case study, $\iota^{\alpha}_{HA}(t)$ is computed as follows.

$$\iota^{\alpha}_{HA}(t) = \frac{\iota^{\alpha}_{HA,1}(t) + \iota^{\alpha}_{HA,2}(t)}{2}. \tag{3.9}$$

Initially, SA does not have any information related to the security system so initial trust levels in this case study are zero as shown in Figure 3.6. Because SA may have different perceived information related to each atomic unit, all atomic units may have different partial trust levels as expected.

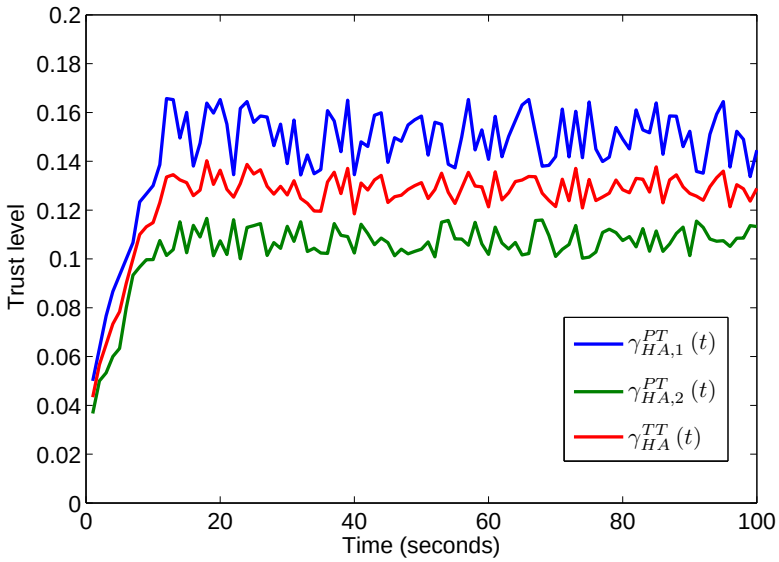

Figure 3.6. Trust levels computed by using perceived information and information extracted from old assessments.

The effects of former assessments are 0.1 and their significance coefficients are 1/3 in this scenario. Therefore, trust levels are always greater than or equal to 0.03 except for $t = 0$.

In the proposed model, SA computes confidences of trust levels optimistically by considering Information Difference among the three information sources. Specifically, the confidence of a trust level is high if the entity has less Information Difference as shown in Figure 3.7. In this scenario, confidences of trust levels are inversely proportional with perceived information if $\iota^{\alpha}_{HA,x}(t) > \iota^{\chi}_{HA,x}(t)$, where $\varphi_x \in \Phi_{HA}(t)$. For

instance, $\iota^{\alpha}_{HA,1}(t) > \iota^{\chi}_{HA,1}(t)$ for $t > 2$ and $\iota^{\alpha}_{HA,5}(t) > \iota^{\chi}_{HA,5}(t)$ for $t > 5$ as shown in Figure 3.7.

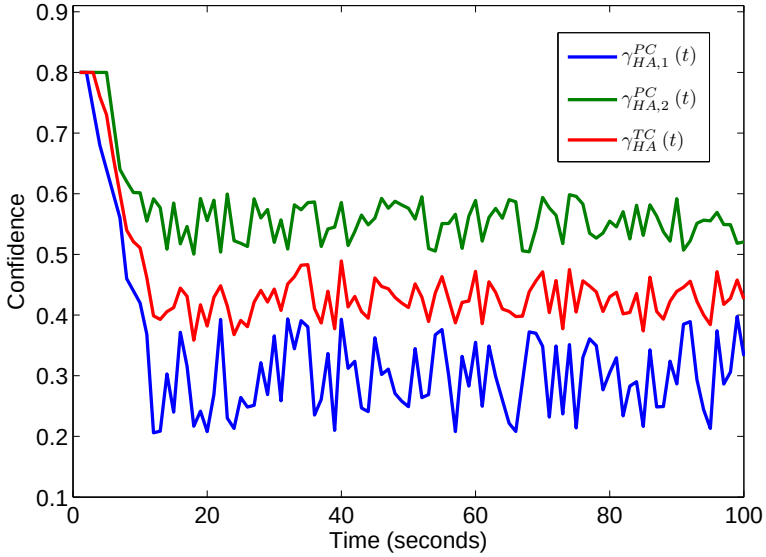

Figure 3.7. Confidences computed by using perceived information and information extracted from old assessments.

Since effects of former assessments to computations of Partial Relative Trust Assessment metrics are constant in this case study, the Partial Relative Trust Assessment metric of an atomic unit depends on its Partial Trust Level and its Partial Confidence as shown in Figure 3.8. Both Partial Trust Level metrics and Partial Confidence metrics change depending on perceived information so that Partial Relative Trust Assessment metric changes only according to perceived information. Additionally, Total Relative Trust Assessment metric is the weighted sum of all Partial Relative Trust Assessment metrics. In this case study, the behavior of $\gamma^{TR}_{HA}(t)$ is more similar to $\gamma^{PR}_{HA,2}(t)$ than $\gamma^{PR}_{HA,1}(t)$ because of Impact Factors, where $\zeta_{HA,1} < \zeta_{HA,2}$.

This scenario shows that an entity may assess the trust of a security system by using only perceived information related to the security system.

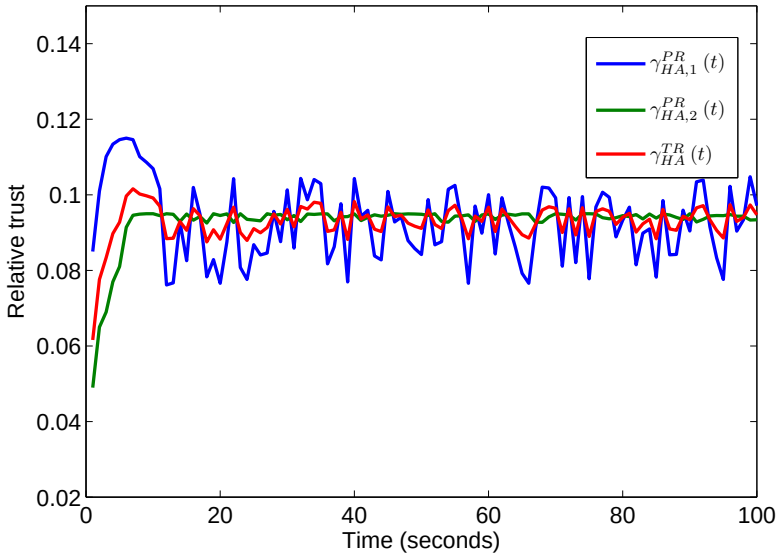

Figure 3.8. Relative trusts computed by using perceived information and information extracted from old assessments.

3.10.2. Scenario 2: Assessments by Using Received Information from Online Appointment System of Hospital A and Information Extracted from Old Assessments

The goal of this scenario is to show effects of information received from a specific service on trust assessments. SA evaluates information received from Online Appointment Service for each atomic unit as shown in Figure 3.9. The service may or may not send information about its security system to SA. Moreover, the service may send incomplete information to SA. For example, the service does not send information to SA for $0 \leq t \leq 5$ as shown in Figure 3.9. The service sends information related to atomic unit ENC, but it does not send information related to atomic unit AC for $31 \leq t \leq 40$. Additionally, SA does not receive information related to atomic unit AC for $61 \leq t \leq 65$.

The service may change the information it sends to SA because of many reasons. For instance, assume that the account management privileges of employees related to the access control to private information of patients is significant for SA. Additionally, assume that there is a rule in the security policy of Hospital A that is "if an employee

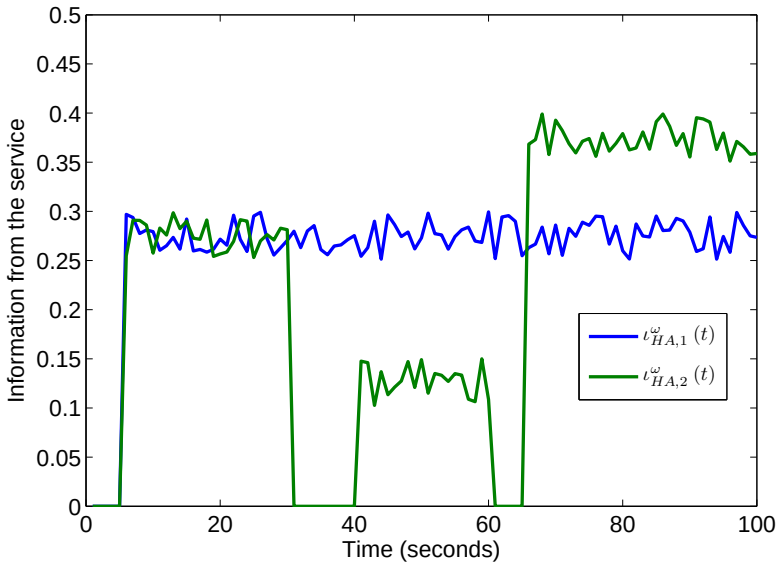

Figure 3.9. Information received from the service evaluated according to needs of the patient.

is fired, her account related to the access control to patients records has to be removed within five days". In this case, SA evaluates information received from the service for $6 \leq t \leq 30$ as shown in Figure 3.9. However, IT department of the hospital is not able to manage accounts of fired employees within five days. Therefore, the hospital updates its security policy, where removing an account of a fired employee is changed to be 10 days. The security policy update decreases the trust to the service in SA for $41 \leq t \leq 60$ as shown in Figure 3.9.

On the other hand, the hospital detects that some fired employees access to private information of some patients if accounts are removed after two days, which may cause problems to the hospital. Therefore, the hospital recruits additional IT specialists and updates its security policy, where the account of a fired employee is removed within one day. Effects of the last update are shown in Figure 3.9 for $66 \leq t \leq 100$. Additionally, similar to $\iota^{\alpha}_{HA}(t)$, the entity evaluates $\iota^{\omega}_{HA}(t)$ as follows.

$$\iota^{\omega}_{HA}(t) = \frac{\iota^{\omega}_{HA,1}(t) + \iota^{\omega}_{HA,2}(t)}{2}. \tag{3.10}$$

In this scenario, a trust level depends only on information received from the service. Therefore, Partial Trust Levels vary according to information related to corresponding atomic unit. For example, $\gamma_{HA,1}^{PT}(t)$ does not change sharply except for $t = 5$ as shown in Figure 3.10. On the other hand, $\gamma_{HA,2}^{PT}(t)$ changes sharply many times because of security policy update of the service. Since information related to the security system is the average of information of atomic units ENC and AC, $\gamma_{HA}^{TT}(t)$ depends on both $\iota_{HA,1}^{\omega}(t)$ and $\iota_{HA,2}^{\omega}(t)$ as shown in Figure 3.10.

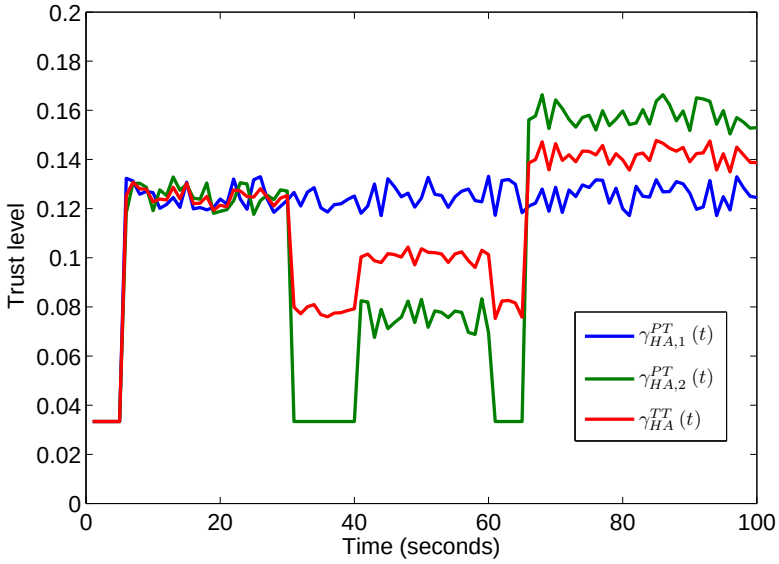

Figure 3.10. Trust levels computed by using information from the service and information extracted from old assessments.

Confidences vary according to information received from the service as shown in Figure 3.11. In this scenario, $iDif_{HA,x}(t)$ and $iDif_{HA}(t)$ depend only on information received from the service, where $\varphi_x \in \Phi_{HA}(t)$. For example, $\gamma_{HA,2}^{PC}(t)$ changes sharply when there are sharp changes in $\iota_{HA,2}^{\omega}(t)$. Moreover, confidences do not exceed 0.8 and trust levels do not fall under 0.1 since we set values of effects of former assessments to be 0.1.

Because $\Psi_{HA,1}(t)$ and $\Psi_{HA,2}(t)$ are constant, Partial Relative Trust Assessments depend only on information received from the service as shown in Figure 3.12. For example, $\gamma_{HA,2}^{PR}(t)$ changes sharply many times whereas $\gamma_{HA,1}^{PR}(t)$ does not change, because of the behavior of information received from the service. Since $\gamma_{HA}^{TR}(t)$ is the

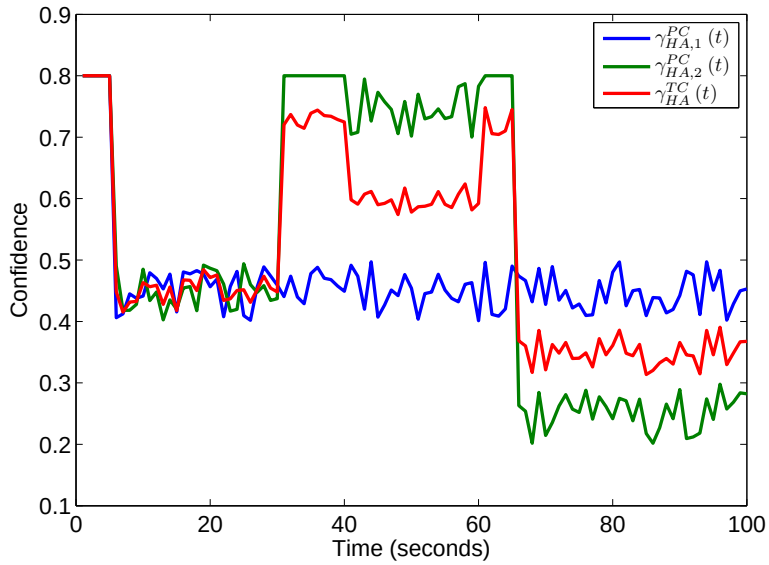

Figure 3.11. Confidences computed by using information from the service and information extracted from old assessments.

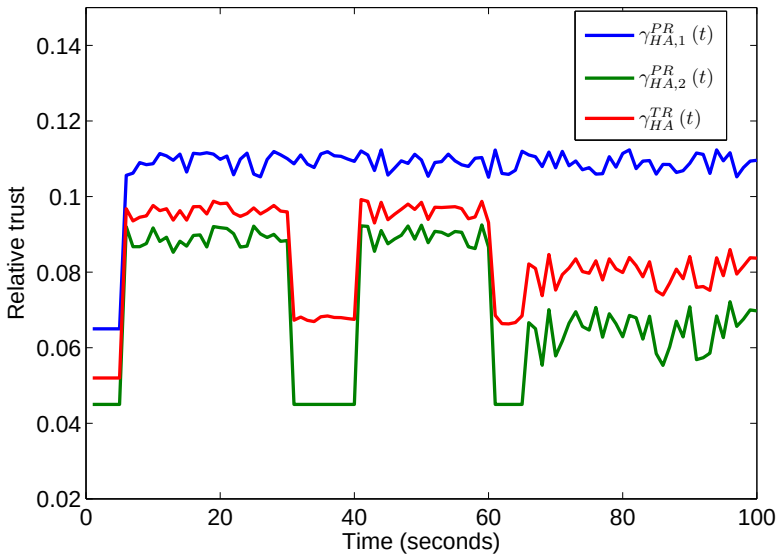

Figure 3.12. Relative trusts computed by using information from the service and information extracted from old assessments.

weighted sum of $\gamma_{HA,1}^{PR}(t)$ and $\gamma_{HA,2}^{PR}(t)$, $\gamma_{HA}^{TR}(t)$ does not change as sharp as $\gamma_{HA,2}^{PR}(t)$. However, $\gamma_{HA}^{TR}(t)$ still reflects the change of the security policy as shown in Figure 3.12.

This scenario shows that an entity can assess the trust of a security system only by using information received from services.

3.10.3. Scenario 3: Assessments by Using All Information

An entity can compute trust metrics more accurately if it has information from the three sources. In this scenario, SA assesses the trust of the security system by using information from these three sources. Specifically, the behavior of perceived information is the same as the perceived information in Scenario 1 and the behavior of information received from the service is the same as information received from the service in Scenario 2. In this scenario, the average values of trust levels are higher than previous two scenarios as shown in Figure 3.13 because SA has totally more information related to the security system.

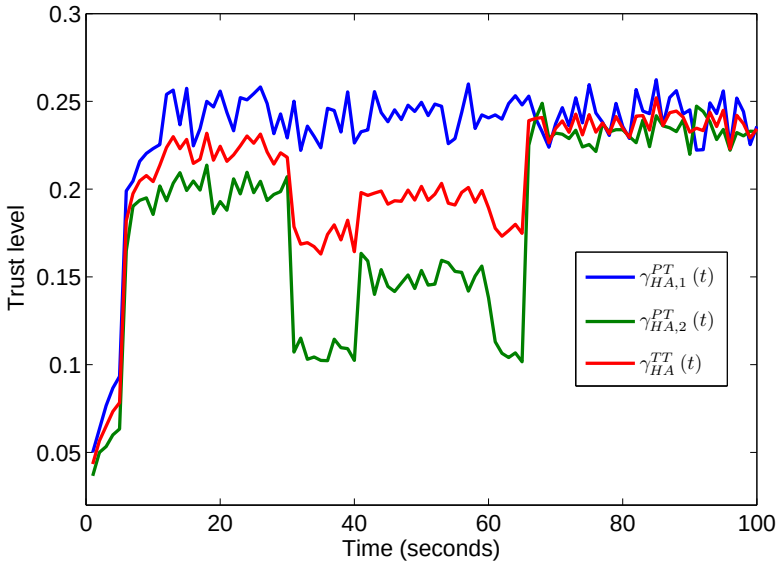

Figure 3.13. Trust levels computed by using all information.

Since SA has information from the three sources, Information Differences vary less than previous two scenarios as shown in Figure 3.14. Therefore, SA has less Information Difference related to an atomic unit so related Partial Trust Level has a

higher Partial Confidence. Figure 3.14 also shows that the number of atomic units is significant for the computation of $\gamma_{HA}^{TC}(t)$ if $\iota_{HA}^{\alpha}(t)$ and $\iota_{HA}^{\omega}(t)$ are computed with Equation 3.9 and Equation 3.10 in sequence. For instance, $\gamma_{HA}^{TC}(t)$ does not change in this scenario as it changes in previous two scenarios.

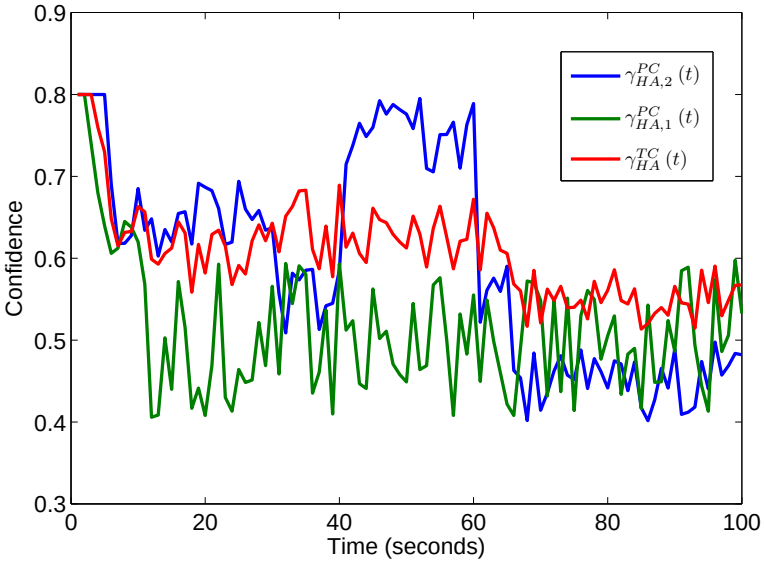

Figure 3.14. Confidences computed by using all information.

Similar to trust level metrics and confidence metrics, the average values of relative trust metrics are higher in this scenario than previous two scenarios as shown in Figure 3.15. Additionally, information variation related to an atomic unit influences Partial Relative Trust Assessment and also Total Relative Trust Assessment. For example, SA does not receive information related to atomic unit φ_2 from the service for $31 \leq t \leq 40$ so $\iota_{HA,2}^{\omega}(t) = 0$. Therefore, both $\gamma_{HA,2}^{PR}(t)$ and $\gamma_{HA}^{TR}(t)$ decrease but $\gamma_{HA,2}^{PR}(t)$ decreases more than $\gamma_{HA}^{TR}(t)$ as shown in Figure 3.15.

This scenario shows that having information related to the security system of a service from three sources ensures trust metrics to be more precise as expected. Therefore, if an entity has information from the three sources, the entity will provide better trust assessments.

The case study shows that the proposed model can be used either with information from one source or with information from all sources. If an entity has information

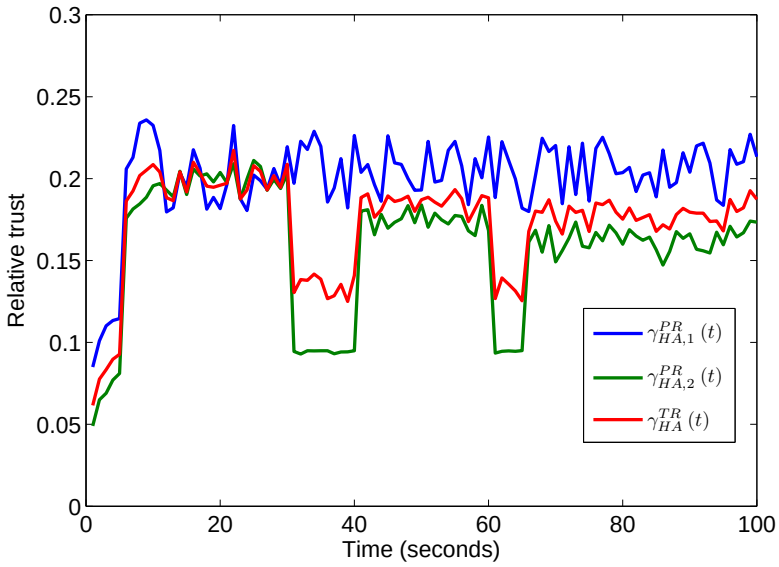

Figure 3.15. Relative trusts computed by using all information.

related to the security system of a service only from one source, the entity can assess the trust of the security system. However, the entity can get better trust assessments if the entity has information from all sources. Briefly, the proposed model is convenient for entities in open environments, where entities are autonomous and the amount of information vary in each entity.

3.11. Conclusion

Emerging open environments are expected to contain autonomous entities and support large number of various services. Such services can be accessible by entities and the entities can be able to interact with each other, where trust to security systems of services becomes a significant issue. In this chapter, we have studied the issue of the trust assessment about the security system of a service according to needs of an entity in an open environment.

We proposed the core model for trust assessment of security systems from entity point of view in open environments. The model contains architecture of Trust Assessment System in an entity. The architecture consists of four layers namely, Communication Interface Layer, Information Classification Layer, Trust Assessment Layer

and Decisions Database Layer. Communication Interface Layer and Decisions Database Layer provide information for assessments while Information Classification Layer manages existing information for next assessment that will carry out on Trust Assessment Layer.

In our model, trust assessments are carried out according to information obtained from three sources. The first source is an entity that perceives information from services and other entities. Perceived information is the combination of information obtained by observations of security systems of services and information received from other entities related to the security system of a specific service. The second source is a service that sends information related to its security system to the entity. The last source is Trust Assessment System of the entity that carries our trust assessments. Trust Assessment System of an entity stores previous trust assessment results and these assessment results contribute to future trust assessments.

The proposed model contains six new trust metrics to assess the security system of a service. Three of these metrics are related to some properties of the security system of a service that we call partial metrics whereas other metrics are used to assess all properties of the security system that we call total metrics. An entity assesses the trust level of a property and the trust level of all properties with Partial Trust Level metric and Total Trust Level metric in sequence. Trust level metrics are computed without considering confidences so that our model has two confidence metrics for the two trust level metrics, namely Partial Confidence metric and Total Confidence metric. Partial Relative Trust Assessment metric and Total Relative Trust Assessment metric are combination of trust level metrics and confidence metrics. Additionally, we presented a step by step trust assessment process that is carried out in Trust Assessment System.

Hospital Online Appointment Service has been presented as a case study with three scenarios. We simulated the scenarios to analyze the proposed trust assessment metrics via different information sources. In the first scenario, trust assessments have been carried out by using perceived information. In the second scenario, trust assessments have been carried out by using information received from a service. In the last

scenario, the entity has both perceived information and information received from a service. The analyses show that an entity that contains a trust assessment system designed according to the proposed model with our new trust assessment metrics can make trust assessments within different conditions in emerging open environments.

4. TRUST EXTRACTION CRUST MODEL: EXTRACTING TRUST INFORMATION FROM SECURITY SYSTEM OF A SERVICE

Entities in open environments have different security needs from services. Trust computations related to the security systems of services necessitate information that meets needs of each entity. Obtaining such information is a challenging issue for entities. In this chapter, we propose a crust model for extracting trust information from the security system of a service based on needs of an entity. We formally represent security policies and security systems to extract trust information according to needs of an entity. The formal representation ensures an entity to extract trust information about a security property of a service and trust information about whole security system of the service. The proposed model is applied to Management Service of Patients' Records of a Dental Clinic as a case study with two scenarios. The scenarios are analyzed experimentally with simulations. The experimental evaluation shows that the proposed model provides trust information related to the security system of a service based on needs of an entity and it is applicable in emerging open environments.

This chapter is organized as follows. We present our motivation and contributions in Section 4.1. We discus related work in Section 4.2. We show formal representation of a security policy and formal representation of a security system in Section 4.3. Section 4.4 is about extracting trust information. In Section 4.5, we present Management Service of Patients' Accounts of a Dental Clinic as a case study to show contributions of the model. We conclude the chapter in Section 4.6.

4.1. Introduction

Having sufficient information is a precondition for making decisions about any property of services in emerging open environments. Emerging open environments are expected to have a large number of services and entities. Entities should obtain

or extract sufficient information for making decisions about services. Depending on the goal of each entity, the amount of sufficient information for making decisions may change. Therefore, entities should obtain information based on their needs.

Since the diversity of services increases in open environments, trust to the security of services has become a significant issue. Security properties in computer science are defined as authentication, confidentiality, integrity, and availability [2, 18, 115]. On the other hand, trust has been investigated in various fields of science, such as philosophy and computer science [1, 57] however there is no agreement about the definition and properties of trust.

Trust to the security system of a service for an entity is a significant problem in open environments. The security system of a service is a set of security mechanisms that are implemented according to the security policy of the service. A security policy is a collection of rules that allow or disallow security related actions and events in a service [3, 7, 8]. On the other hand, a security mechanism implements security policies in the system.

4.1.1. Motivation

An entity should trust to security systems of services to interact with them in emerging open environments. Therefore, assessing the trust of the security system of a service according to needs of an entity is becoming a significant issue. Additionally, each entity needs a trust assessment model and information for making trust assessments related to the security system of a service.

In literature, there are many trust computation models that can be applied for assessing the trust of the security system of a service. On the other hand, obtaining information related to the security system of a service according to needs of an entity is not clearly addressed. One entity may gather information for trust computations from other entities and services. The entity may also extract information directly from the security system of a service.

Existing trust models do not provide a solution to extract trust information related to the security system of a service according to needs of a specific entity. Our motivation is the lack of a model for extracting trust information from the security system of a service based on needs of a specific entity in open environments.

4.1.2. Contributions

We propose a crust model for extracting trust information from the security system of a service in emerging open environments. Trust information is extracted from the security system of a service based on needs of an entity, the security policy of the entity, and the security policy of the service. The proposed model has been applied to the security system of a management service of patients' account as a case study. The proposed model has been evaluated experimentally with simulations in the case study. We can summarize the contributions of our work as follows.

- We represent the security policy of an entity and the security policy of a service with sets of atomic units according to needs of the entity. The set representation of security policies provides a way to demonstrate the needs of a specific entity from the security system of a particular service in emerging open environments.

- We propose a novel model for extracting trust information from the security system of a service. The model considers needs of an entity from the security system of a specific service. An entity can extract trust information related to a specific security property and the whole security system of a service by using the model.

4.2. Related Work

Contemporary services are highly dynamic and untrustworthy. Sensitive data of the services are continuously exposed to the risk of being delivered to final users. Intermediary actors who do not have access rights to data are taking part to data transactions. An approach to manage data privacy for data exchange is proposed

in [108]. The approach considers the front-end trust filter paradigm. The paradigm aims to guarantee high flexibility, reduce the resources required, and limit pervasiveness into applications and devices.

The problem regarding security properties of communicating agents is analyzed in [111]. Temporal belief logic is used to show how to establish dynamic trust theories for communication protocols. A trust theory for a given security mechanism of communication systems is presented and a global assumption set is construct. Both the trust theory and the global assumption set are used to show a security property. In another research, an information gain metric is used to dynamically extract tendencies of failure of target agents [116]. Autonomic trust extraction is also studied for trustworthy service discovery in urban computing in [117]. The value of trust is automatically determined according to interactions between a user and a service.

Trust Computing Group (TCG) presents a secure computing environment and a testing prototype to solve trust problems of the secure computing environment [118]. Specifically, the prototype intends to eliminate the gap between TCG specifications and product implementations. An automata theory is introduced as a test mechanism to achieve TPM specification compliance test, validate chain of trust compliance by analyzing TCG-BIOS, and use reflection mechanism to test each layer of TSS. The significance of this research is that it divides the entire system into pieces and begins to calculate trust according to these pieces.

Ad-hoc networks are another popular context for the trust research. For instance, trust metrics related to ad-hoc networks are evaluated in [112]. An example related to trusted routing in mobile ad-hoc networks is presented in [113]. A more specific study that considers essentials for developing a good trust management system for wireless sensor networks is presented in [119].

Although these researches contain many contributions to trust, none of them presents a solution that describes the way to extract trust information from the security system of a service based on the needs of a specific entity. In our model, each entity

can extract trust information from the security system of a service based on its current needs from that service.

4.3. Security Policy and Security Mechanism

A *security policy* is a collection of rules that allow or disallow possible actions, events, or something related to the security of an entity or a service. A *security system* is a set of security mechanisms. The security system of a service is a set of security mechanisms enforced according to the security policy of the service. A *security mechanism* of a service is an application to enforce rules of a security policy or could also be interpreted as a security property.

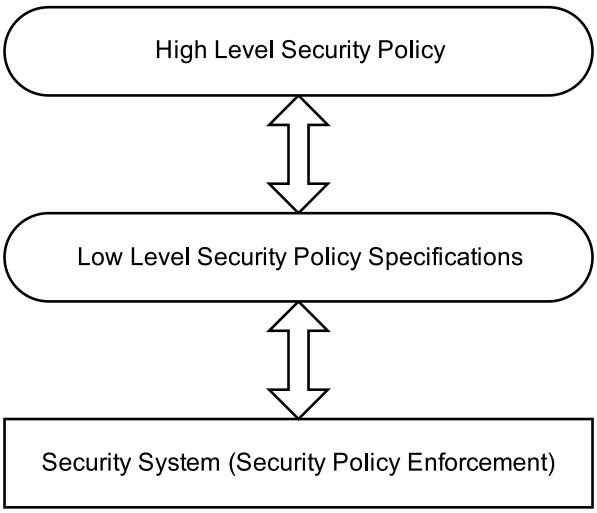

Figure 4.1. Security policy enforcement hierarchy.

Security policies have different granularities. High level security policies are close to natural languages and they are sometimes represented with formal natural languages. Briefly, high level security policy languages are derivatives of natural languages that are modified according to needs of an application. On the other hand, low level security policies are formal specifications of high level security policies. Therefore, low level security policy specifications are expected to be enforced directly to systems. Figure 4.1 shows general security policy enforcement hierarchy.

Security policy of an entity represents its security needs from the security system of a service. However, the security system of a service is an enforcement of the security

policy of the service. Therefore, the security system of a service may not comply with the security policy of an entity. The relation between the security policy of an entity and the security system of a service is shown in Figure 4.2a. Briefly, the security system of a service is an enforcement of the security policy of the service, but an entity actually needs to determine the degree the security system enforces its security policy.

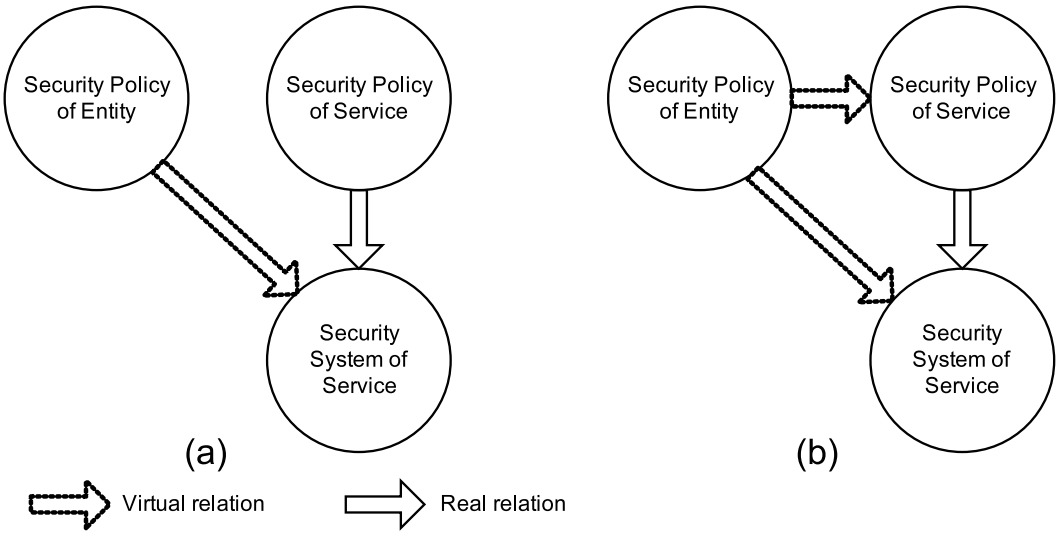

Figure 4.2. Relations among the security policy of an entity, the security policy of a service, and the security system of the service according to the entity point of view (a) expected relations (b) real relations.

The security system of a service may not apply all features of the security policy of the service. The amount of the enforcement is an indicator of trust for an entity so the amount of the enforcement is significant to extract information for trust computations.

The security policy of a service represents the expected behavior of the security system of the service from the service point of view. Additionally, the security policy of an entity represents expected security relations of the entity with services. Therefore, the relation between the security policy of an entity and the security policy of a service is a part of information for trust computations as shown in Figure 4.2b.

High level security policies of entities and services have to be represented formally according to needs of each specific entity to extract trust information. Additionally, the security system of a service has to be represented in a formal manner. In our model,

an entity represents security policies and security mechanisms with sets of atomic units according to its own needs.

In the proposed model, each entity has access to security policies and security systems of services. Specifically, services send their security policies to entities on the request of the entities. Entities then process and represent the received security policy and the security system of a service according to their own needs. Therefore, each entity may have different representations of security policies and security systems.

Entities and services may have different security policies depending on their specific needs. For example, some entities and services may have access control policies while some others may have communication security policies. On the other hand, some other entities and services may have both access control policies and communication security policies.

4.3.1. Formal Representation of Security Policy by Atomic Units

We represent a security policy with atomic units according to needs of an entity from a specific service. An *atomic unit* may be a rule of a security policy or a set of rules of the security policy. For instance, assume that the security policy of an entity has two rules that define the expected access control behavior of a service with which services the entity interacts. The first rule defines access control to a service from intranet and the second rule defines access control from the Internet, where intranet is expected to have more trustworthy entities than the Internet has. Therefore, the first rule necessitates more strict access control than the second rule. If the trustworthiness of intranet differs from the trustworthiness of the Internet on an entity, the entity represents two rules with separate atomic units. On the other hand, another entity may not distinguish the Internet and intranet so the entity may represents the two rules with a single atomic unit.

Set $P_c^s(t) = \{p_1, \ldots, p_m\}$ represents the security policy of service ω_c in an entity, where $m \in \mathbb{Z}^+$ and p_j denotes an atomic unit of set $P_c^s(t)$. A security policy of an entity

is also represented with a set of atomic units in the entity. Set $P_c^e(t) = \{p_1, \ldots, p_n\}$ represents the security policy of an entity related to service ω_c in that entity, where $n \in \mathbb{Z}^+$ and p_k denotes an atomic unit of set $P_c^e(t)$.

Assume that the security policy of an entity has two rules related to security expectations from the online ticked reservation service of an airlines. The first rule is *Passengers' information are encrypted and then stored*, whereas the second rule is *All encryptions are carried out on a Trusted Platform Module on the service*. Additionally, assume that the entity represents the first rule with atomic unit *CRYPTO* and the second rule with atomic unit *TPM*. In this case, the set representation of the security policy of the entity is $P_{air}^e(t) = \{CRYPTO, TPM\}$. On the other hand, assume that the security policy of an airline online ticked reservation service has the first rule only so the security policy of the service is represented with $P_{air}^s(t) = \{CRYPTO\}$. Note that security policies may be dynamic and needs of an entity may change in course of time so the elements of a set may change with time.

Either an entity or a service may have complex security policies. The complexity of a security policy depends on the amount of rules and dependencies among the rules. An entity represents only some parts of its security policy, which are related to services depending on its needs from that services. For example, the security policy of an entity may contain rules that are about relationships with other entities, which rules are not related to services. Therefore, the entity does not represent these rules to extract trust information.

In this chapter, an entity has one security policy that security policy defines all requirements of the entity with rules. Similarly, a service has a unique security policy that defines requirements of the service with rules from its security system.

4.3.2. Formal Representation of Security System by Atomic Units

In this chapter, the atomic unit representation of a security system is the same as the atomic unit representation of the security system of a service described in Section

3.5. Specifically, we represent the security system of a service from the entity point of view with atomic units. Each entity can extract information about all atomic units of the security system of a specific service. Because the security system of a service is a set of security mechanisms, the atomic unit representation of the security system of a service is an atomic unit representation of security mechanisms according to needs of a specific entity.

An *atomic unit* can be a property of a security mechanism or some properties of the security mechanism. Additionally, an atomic unit can be a security mechanism or a set of security mechanisms. For instance, assume that the security system of a service has a password based authentication mechanism and a digital certificate authentication mechanism. The password based authentication mechanism has two properties namely, the minimum length of a password constraint and the password content constraint. An entity may represent the length of a password constraint and the content of a password constraint with different atomic units. On the other hand, another entity may represent the password based authentication mechanism with a single atomic unit. Moreover, some other entities may represent both the password based authentication mechanism and the digital signature based authentication mechanism with one atomic unit.

4.4. Extracting Trust Information

An entity expects that the security policy of a service is the same as with its security policy. The entity has rules in its security policy that describe its security needs from the security system of a service. A service also has rules in its security policy that describe its needs from its security system. Briefly, the security policy of a service may differ from the security policy of an entity. If the security policy of service ω_c is different from the security policy of an entity and the entity needs to obtain information from that service for trust computations, the security policy of the service has fewer rules than the security policy of the entity. Formally, $|P_c^e(t)| > |P_c^s(t)|$.

4.4.1. Expected Sets

The rules of the security policy of an entity that represent needs of the entity from the security system of a service are a subset of all rules of the entity's security policy. Similarly, the rules of the security policy of a service that are related to needs of a specific entity may be a subset of all rules of the security policy of the service.

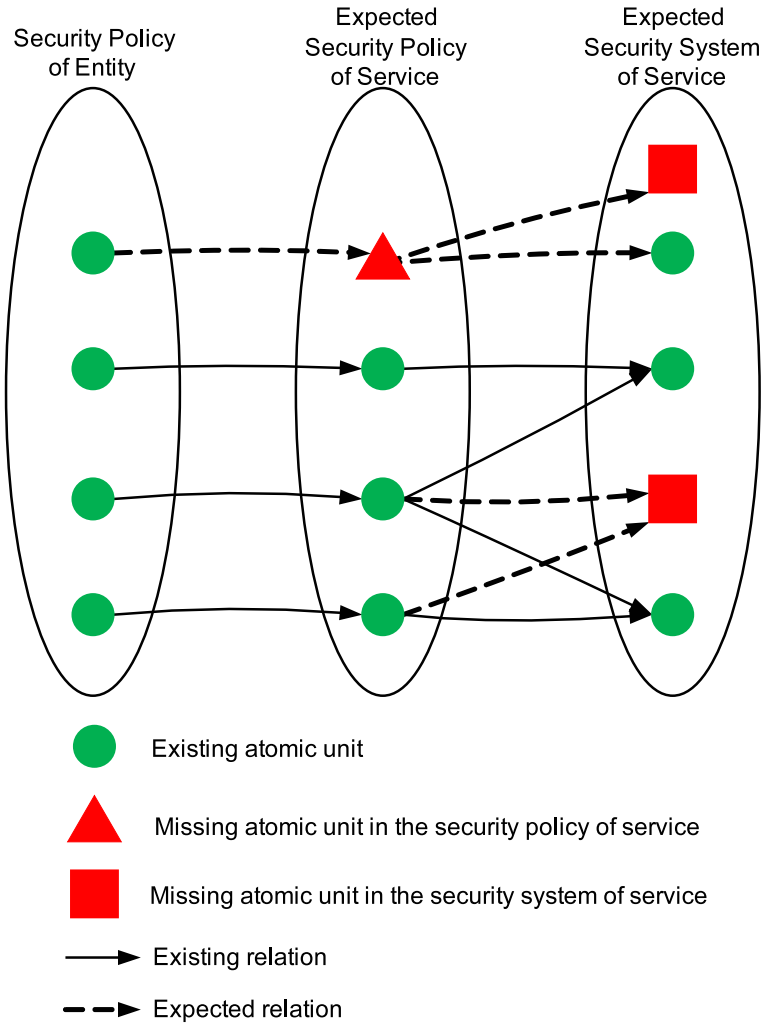

Figure 4.3. Atomic relations among the security policy of an entity, the expected security policy of a service and the expected security system of the service.

Although an entity expects the security policy of a service to be the same with its security policy, the security policy of the service is usually different from the security policy of the entity. Therefore, an entity has an *expected security policy* related to a specific service. The expected security policy related to service ω_c is represented with set $P_c^{xs}(t)$. Set $P_c^{xs}(t)$ has equal number of members with set $P_c^{e}(t)$ that means

$|P_c^e(t)| = |P_c^{xs}(t)|$.

Actually, an entity uses binary relations between set $P_c^e(t)$ and set $P_c^s(t)$ to extract trust information. Therefore, missing atomic units are used to complete the security policy of a service as shown with triangles for set $P_c^{xs}(t)$ in Figure 4.3. A *missing atomic unit* stands for an absent rule in the security policy of the service. Missing atomic units do not exist in set $P_c^s(t)$.

The security system of a service may not satisfy its security policy. Because the security policy of a service is dynamic, the security system of the service is also expected to be dynamic. However, the security system of a service may not be updated immediately when the security policy is updated. Additionally, the security system may be implemented incorrectly or it may be incomplete. Therefore, the security system may not represent correct enforcement of its security policy. In other words, set $\Phi_c(t)$ may not have atomic units that are needed to represent all relations between atomic units of set $\Phi_c(t)$ and atomic units of set $P_c^s(t)$. Moreover, set $\Phi_c(t)$ may not have atomic units that are needed to represent all relations between atomic units of set $\Phi_c(t)$ and atomic units of set $P_c^{xs}(t)$. Therefore, an entity has the expected security system of service ω_c that meets requirements for the representation of relations between atomic units of set $\Phi_c(t)$ and atomic units of set $P_c^{xs}(t)$.

The *expected security system* of service ω_c is represented with $\Phi_c^x(t)$ and the relations between set $P_c^{xs}(t)$ and set $\Phi_c^x(t)$ are shown in Figure 4.3. Similar to the expected security policy representation of a service, we use missing atomic units to complete set $\Phi_c^x(t)$. A *missing atomic unit* in set $\Phi_c^x(t)$ is represented with a square as shown in Figure 4.3. Note that set $\Phi_c(t)$ does not contain missing atomic units.

An atomic unit of set $P_c^{xs}(t)$ may be related to one atomic unit or more than one atomic unit of set $\Phi_c^x(t)$ as shown in Figure 4.3. Moreover, each atomic unit of set $\Phi_c^x(t)$ may have different weights on an atomic unit of set $P_c^{xs}(t)$. The *weight of atomic unit* $\varphi_i \in \Phi_c^x(t)$ on atomic unit $p_j \in P_c^{xs}(t)$ is represented with $w_{j,i}(t)$, where $w_{j,i}(t) \in [0,1]$ and $w_{j,i}(t)$ satisfies the following condition.

$$\sum_{\forall \varphi_i \in \Phi_c^x(t)} w_{j,i}(t) = 1,\ p_j \in P_c^{xs}(t). \tag{4.1}$$

If there is no relation between two atomic units, then $w_{j,i}(t) = 0$. The value of $w_{j,i}(t)$ represents the significance of atomic unit φ_i related to atomic unit p_j for extracting trust information. For instance, if $w_{j,i}(t) = 0.6$ and $w_{j,r}(t) = 0.4$, then atomic unit φ_i is more significant than atomic unit φ_r to extract trust information related to atomic unit p_j, where $\varphi_i, \varphi_r \in \Phi_c^x(t)$. Additionally, if atomic unit p_j is related only to one atomic unit, such as to atomic unit φ_i, then $w_{j,i}(t) = 1$.

4.4.2. Satisfaction Factor

An atomic unit of a security system may not satisfy its specification fully because of many reasons, such as implementation problems. For example, assume that an atomic unit of the security system of a service specifies an acceptable password for the authentication mechanism of the service, where a password has to contain at least two capital letters and the password length has to be at least eight characters. If a service allows an entity to determine a password with less than two capital letters but does not allow determining the password length to be less than eight characters, the atomic unit of the security system partially satisfies its specifications.

The *satisfaction factor of an atomic unit of a security system* represents the satisfaction ratio of the atomic unit based on needs of a specific entity. The satisfaction factor related to the expected security system of service ω_c is represented with $sts_i(t)$, where $sts_i(t) \in [0, 1]$ and $\varphi_i \in \Phi_c^x(t)$. If atomic unit φ_i fully satisfies its specifications, then $sts_i(t) = 1$. On the other hand, if atomic unit φ_i does not satisfies any of its specifications or it is a missing atomic unit, $sts_i(t) = 0$. Otherwise, $0 < sts_i(t) < 1$. A satisfaction factor depends on needs of an entity and the security policy of a service on which the security policy is implemented. Therefore, the value of a satisfaction factor

is dynamic.

The *satisfaction factor of atomic unit* shows how much the security policy of service ω_c satisfies the rule in the expected security policy of the service. We represent the satisfaction factor of atomic unit $p_j \in P_c^{xs}(t)$ with $stp_j(t) \in [0,1]$. Satisfaction factor $stp_j(t)$ is the weighted sum of satisfaction factors of corresponding atomic units in $\Phi_c^x(t)$, which is computed as follows.

$$stp_j(t) = \sum_{\forall \varphi_i \in \Phi_c^x(t)} w_{j,i}(t)\, sts_i(t)\,,\ p_j \in P_c^{xs}(t)\,. \tag{4.2}$$

4.4.3. Histories

The security policy of a service and its corresponding security system may change with time. These changes provide information for trust computations. An entity may have more trust to the security system of a service if the security system is improved according to needs of the entity. On the other hand, a change in the security policy of a service or a change in the security system of the service may result in lower trust to the security system on the entity. Specifically, atomic units of the security policy of a service and atomic units of the security system of the service may change with time. Histories of these changes provide information for trust computations.

The *history of atomic unit* $\varphi_i \in \Phi_c^x(t)$ represents changes of the atomic unit in relation to atomic unit $p_j \in P_c^{xs}(t)$ according to needs of a specific entity related to service ω_c. We represent the effect of history of atomic unit $\varphi_i \in \Phi_c^x(t)$ with $h_{j,i}^{sss}(t)$, where $p_j \in P^{xs}(t)$ and $h_{j,i}^{sss}(t) \in [-1,1]$. The history effect may be positive or negative and it changes with time. Moreover, all history effects that are related to atomic unit p_j are combined as following, where $h_j^{sss}(t)$ represents the combined histories of atomic units in $\Phi^x(t)$ that is related to atomic unit p_j.

$$h_j^{sss}(t) = \sum_{\forall \varphi_i \in \Phi_c^x(t)} h_{j,i}^{sss}(t),\ p_j \in P_c^{xs}(t). \tag{4.3}$$

Similarly to histories of atomic units of set $\Phi_c^x(t)$, an atomic unit of set $P_c^{xs}(t)$ has a history that represents changes in that atomic unit according to needs of a specific entity. We represent the effect of the history of atomic unit $p_j \in P_c^{xs}(t)$ with $h_j^{ssp}(t) \in [-1, 1]$.

Since atomic units of set $\Phi_c(t)$ are enforcements of atomic units of set $P_c^s(t)$, an atomic unit of set $P_c^s(t)$ is also affected by histories of related atomic units of set $\Phi_c(t)$. Therefore, the *overall history effect* on atomic unit $p_j \in P_c^{xs}(t)$ depends on histories of related atomic units of set $\Phi_c^x(t)$ and the history of the atomic unit of set $P_c^{xs}(t)$. The overall history effect related to atomic unit p_j is represented with $h_j(t) \in [-1, 1]$ and is computed as below.

$$h_j(t) = \min\left(1, \max\left[-1, h_j^{sss}(t) + h_j^{ssp}(t)\right]\right). \tag{4.4}$$

The combination of the overall history effect and the weighted sum of satisfaction factors of an atomic unit in set $P_c^{xs}(t)$ is the information extracted for trust computations from the security policy of a service and related security system. We call *trust information of an atomic unit* in set $P_c^{xs}(t)$ for such information. Trust information related to atomic unit $p_j \in P_c^{xs}(t)$ is represented with $ta_j(t) \in [0, 1]$ and is computed as following.

$$ta_j(t) = \begin{cases} 1 & , stp_j(t) + h_j(t) > 1 \\ 0 & , stp_j(t) + h_j(t) < 0 \\ stp_j(t) + h_j(t) & , otherwise \end{cases} \quad (4.5)$$

4.4.4. Perception Factor

An entity can extract trust information related to a specific atomic of its set $P_c^e(t)$ or all atomic units of the set according to its present needs from the security system of service ω_c. In our model, each atomic unit of set $P_c^e(t)$ depends on an atomic unit of set $P_c^s(t)$. Therefore, trust information of an atomic unit of set $P_c^e(t)$ depends on the trust information of an atomic unit of set $P_c^{xs}(t)$. Moreover, each entity can perceive trust information of an atomic unit of set $P_c^{xs}(t)$ differently so entities have a perception factor for each atomic unit of set $P_c^e(t)$. The *perception factor* shows the effect of atomic unit $p_j \in P_c^{xs}(t)$ to atomic unit $p_k \in P_c^e(t)$ and is represented with $\pi_{k,j}(t) \in [0,1]$. If atomic unit p_j fully affects atomic unit p_k, $\pi_{k,j}(t) = 1$. If $\pi_{k,j}(t) = 0$, atomic unit p_j does not affect atomic unit p_k or atomic unit p_j is a missing atomic unit. Briefly, a perception factor shows the belief of an entity to information received from a service related to a specific atomic unit of set $P_c^e(t)$. A perception factor is private to an entity so it may differ from one entity to another one.

4.4.5. Impact Factor

The *impact factor* of an atomic unit in set $P_c^e(t)$ shows how much the extracted trust information related to the atomic unit contributes to all extracted trust information. The impact factor of atomic unit $p_k \in P_c^e(t)$ is represented with $imp_k(t) \in [0,1]$ and it satisfies the condition shown with Equation 4.6. If $imp_k(t) = 1$, extracted trust information related to atomic unit p_k has maximum impact to the computation of all extracted trust information. Whereas, if $imp_k(t) = 0$, extracted trust information related to atomic unit p_k has no impact.

$$\sum_{\forall p_k \in P_c^e(t)} imp_k(t) = 1. \tag{4.6}$$

4.4.6. Trust Information Metrics

Extracted trust information related to an atomic unit in set $P_c^e(t)$ consists of all trust information extracted from the security system of a service based on needs an entity. Specifically, *extracted trust information* is a combination of information extracted from the security policy of a service and the security system of the service. Additionally, it depends on the perception of the entity. Extracted trust information from service ω_c in an entity related to atomic unit $p_k \in P_c^e(t)$ is represented with $\iota_k(t) \in [0,1]$ and is computed as following.

$$\iota_k(t) = \pi_{k,j}(t)\, ta_j(t),\ p_k \in P_c^e(t), p_j \in P_c^{xs}(t). \tag{4.7}$$

While extracted trust information related to an atomic unit of set $P_c^e(t)$ shows the trustworthiness of the atomic unit, extracted trust information related to all atomic units of set $P_c^e(t)$ shows the trustworthiness of the security system of service ω_c. Extracted trust information related to all atomic units is a weighted sum of extracted trust information related to each atomic unit of set $P_c^e(t)$. We represent extracted trust information related to all atomic units of set $P_c^e(t)$ according to needs of an entity from the security system of service ω_c with $\iota(t) \in [0,1]$. Extracted trust information related to all atomic units is computed as following.

$$\iota(t) = \sum_{\forall p_k \in P_c^e(t)} imp_k(t)\, \iota_k(t). \tag{4.8}$$

The value of $\iota_k(t)$ shows the trustworthiness of atomic unit $p_k \in P_c^e(t)$ and the value of $\iota(t)$ shows the trustworthiness of the security system of service ω_c based on needs of an entity. For instance, $\iota_k(t) = 1$ means that atomic unit p_k has maximum trustworthiness on the entity. On the other hand, if $\iota(t) = 0$, the security system of the service is not trustworthy according to needs of a specific entity.

Consequently, our contribution is to show a way to extract trust information for complicated trust computations. Trust information is extracted from the security system of a service based on the needs of a specific entity. The security policy of the entity and the security policy of the service contribute to extract trust information. We have two types of trust information. The first type of information is related to a specific security property of a service, whereas, the second type is related to all security system of the service.

4.5. Case Study: Dental Clinic Patient Service

We have simulated the proposed model with a case study and have conducted several experiments. The case study and experiments have two objectives. The first objective is to illustrate the applicability of the proposed model on possible applications. The second objective is to show the effects of changes in a security policy and in a security system to extracting trust information. Therefore, the case study is about extracting trust information from patients' records management service of a dental clinic according to needs of a person. Experiments are conducted with two scenarios. In the first scenario, we have analyzed the effect of changes in security system of a service. In the second scenario, we have investigated effects of changes in the security policy of the service and in the security policy of an entity.

We simulated the scenarios and showed the performance results based on our proposed model. Simulations were carried out by using MATLAB R2009b version 7.9.0.529 that run on a PC with Intel Core 2 Duo E8400 3.00GHz processor and 3GB of RAM.

4.5.1. Case Study Overview

Suppose that security policies and security systems of dental clinics are public for potential patients who wish to get appointments from the Internet. Suppose also that a patient has dental problems and she needs to get an appointment from a dental clinic close to her location. There are many dental clinics close to the patient's location. Dental clinics store patients' records on data storages that are accessible from the Internet. However, the patient knows that some clinics have weak security systems of their patients' record management service. Therefore, medical records may be revealed by an adversary.

Medical privacy is significant for the patient so that a dental clinic has to keep patients' medical records from being revealed to other people. If medical records of the patient are revealed, the personal life of the patient will be affected. Moreover, the patient may have financial problems, such as increasing of insurances costs. On the other hand, the patient has some positive recommendations about a dentist in dental clinic BDENT. Before getting an appointment from the dentist in BDENT, the patient needs to assess the trust of the security system of Patients' Records Management Service of BDENT.

The patient has a software agent (PA) that represents herself on the Internet, where the software agent is the entity in this case study. The software agent can get information related to Patients' Records Management Service of BDENT and it can assess the trust of the security system of the service according to privacy needs of the patient. Then, the patient can decide whether to get an appointment or not by considering the trust assessments.

The *security policy of entity PA* has the following rules related to the security system of a service:

(i) Patients' records have to be stored in an encrypted form and the encryption keys have to be kept secure in a hardware device.
(ii) Dentists have to access only to their patients' records by using password based authentication, where passwords are stored in an encrypted form.
(iii) The security system of a service has to contain a monitoring mechanism for auditing all accesses to patients' records.

According to the rules, PA represents its security policy with three atomic units, where each atomic unit corresponds to a rule in the security policy. The first rule is represented with atomic unit ECE. The second and the third rules are represented with atomic units ATE and ADE respectively. Therefore, the set representation of the security policy of entity PA is $P^{e}_{BDENT}(t) = \{ECE, ATE, ADE\}$.

The security policy of BDENT has following rules related to Patients' Records Management Service:

(i) Patients' records are stored in an encrypted form.
(ii) Dentists access only to their patients' records by using password based authentication, where passwords are stored in an encrypted form.

PA knows the security policy of BDENT because the security policy is public. PA represents the security policy with two atomic units, where ECS corresponds to the first rule and ATS corresponds to the second rule. In this case, the set representation of the security policy of BDENT is $P^{s}_{BDENT}(t) = \{ECS, ATS\}$. Normally, PA expects to see a rule in the security policy of BDENT that is related to the third rule in its security policy, but the security policy does not contain such a rule. Therefore, PA has a missing atomic unit in $P^{xs}_{BDENT}(t)$ that is related to the third rule of its security policy. The missing rule is represented with ADS so $P^{xs}_{BDENT}(t) = \{ECS, ATS, ADS\}$.

The security system of BDENT has a password based authentication mechanism. Both passwords and patients' records are encrypted with an encryption mechanism. The encryption mechanism uses AES algorithm for encryptions. However, encryption keys are not stored in a hardware device. Additionally, the security system does not contain any monitoring mechanism for logging accesses to patients' records. Therefore, the set representation of the security system of BDENT has two atomic units. Atomic unit AES represents the encryption mechanism whereas atomic unit PW represents the password based authentication mechanism. Atomic unit TPM represents a possible hardware for storing encryption keys and atomic unit LOG represents a possible monitoring mechanism of the security systems. Atomic units TPM and LOG of the security system are missing atomic units in this case so the set representation of the security system is $\Phi_{BDENT}(t) = \{AES, PW\}$ whereas the set representation of the expected security system is $\Phi^{x}_{BDENT}(t) = \{AES, PW, TPM, LOG\}$. Relations among the sets in PA are show in Figure 4.4.

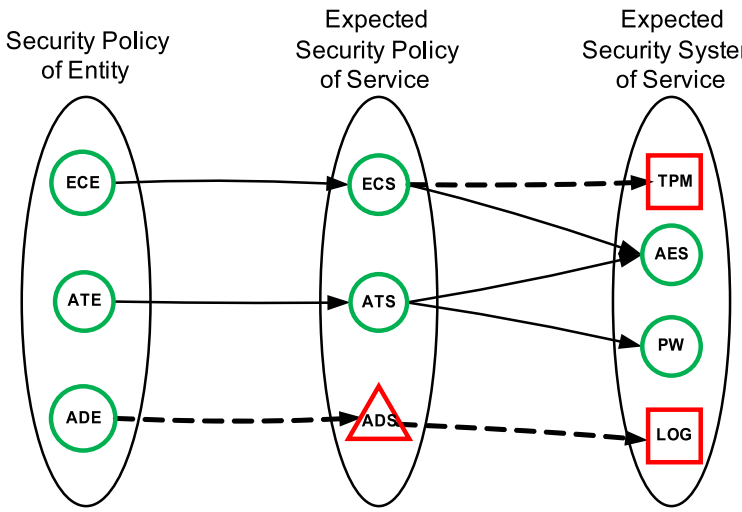

Figure 4.4. Atomic relations among the sets in PA related to BDENT.

4.5.2. Scenario 1: Effects of Changes in Security System

In this scenario, we examine impacts of changes in the security system of a service related to extracted trust information. Specifically, we update some security mechanisms of the security system of BDENT and show effects of the update. Moreover, we analyze effects of $sts_i(t)$, such that $\varphi_i \in \Phi^{x}_{BDENT}(t)$.

Because we are interested in effects of $sts_i(t)$ and changes in the number of atomic units of the security system in this scenario, we chose some parameters to be constant. We assume that effects of histories are zero, $h^{sss}_{j,i}(t) = 0$ and $h^{ssp}_{j}(t) = 0$, where $p_j \in P^{xs}_{BDENT}(t)$ and $\varphi_i \in \Phi^{x}_{BDENT}(t)$. The perception factors are $\pi_{ECE,ECS}(t) = 0.8$, $\pi_{ATE,ATS}(t) = 1$ and $\pi_{ADE,ADS}(t) = 0$. The impacts of each atomic unit to all extracted trust information are $imp_{ECE}(t) = 0.3$, $imp_{ATE}(t) = 0.5$ and $imp_{ADE}(t) = 0.2$. The weight factors of atomic units are $w_{ECS,TPM}(t) = 0.3$, $w_{ECS,AES}(t) = 0.7$, $w_{ATS,AES}(t) = 0.4$, $w_{ATS,PW}(t) = 0.6$, and $w_{ADS,LOG}(t) = 1$.

Normally, it is expected that the satisfaction factor of an atomic unit does not change unless there are changes in the related security mechanisms. However, needs of entities may change so satisfaction factors are expected to vary with time. Therefore, values of $sts_{AES}(t)$ vary between 0.85 and 0.95 in this scenario as shown in Figure 4.5.

Initially, the minimum password length has to be at least eight characters in BDENT so that values of $sts_{PW}(t)$ are between 0.65 and 0.75 for $0 < t < 10$. However, the minimum password length is changed to be at least four characters at $t = 10$ because some patients do not remember their passwords and have to contact to the security management desk, which circumstance brings additional cost to BDENT. However, IT management department of BDENT observes that patients have low trust to services with short password lengths therefore the minimum password length is updated to be at least six characters at $t = 20$. The security system of BDENT is updated according to these changes. PA also changes values of the satisfaction factor related to PW to be between 0.25 and 0.35 for $10 < t < 20$ and between 0.45 and 0.5 for $t > 19$ as shown in Figure 4.5. Additionally, the security policy of BDENT and the security system are updated simultaneously according to these changes.

Actually, the security system of BDENT has monitoring software that logs accesses to patients' records. However, the monitoring software has been turned off until $t = 10$. When the required length of passwords are updated at $t = 10$, the monitoring software is turned on. On the other hand, the security policy of BDENT is never updated so PA does not consider this change and $\iota_{ADE}(t) = 0$ as shown in Figure

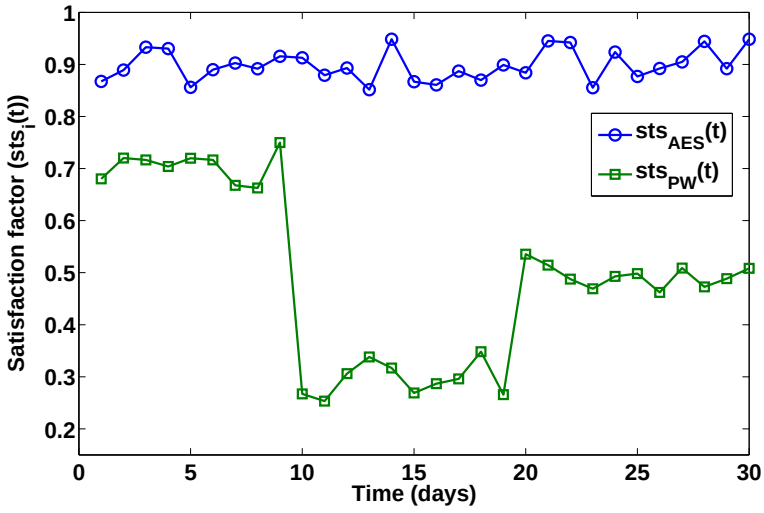

Figure 4.5. The change of satisfaction factors when the security system of BDENT is changed.

4.6. Briefly, changes in a security system of a service do not affect the extracted trust information if the changes do not have corresponding rule in the security policy of the service.

In this scenario, extracted trust information varies for some atomic units as shown in Figure 4.6. Since perception factor $\pi_{ADE,ADS}(t) = 0$, extracted trust information $\iota_{ADE}(t)$ is zero. On the other hand, extracted trust information related to other atomic units varies all the time. However, the variance of $\iota_{ATE}(t)$ is greater than the variance of $\iota_{ECE}(t)$ when the security system is updated. For example, the atomic unit PW is updated at $t = 10$ and $t = 20$, where $\iota_{ATE}(t)$ changes considerably. $\iota_{ECE}(t)$ does not vary as much as $\iota_{ATE}(t)$ because there is no change in atomic units AES and TPM in set $\Phi_{BDENT}(t)$. These examples show that extracted trust information related to an atomic unit of a security system depends highly on updates of the atomic unit.

Extracted trust information related to all security system of BDENT depends on impact factors and the extracted trust information related to each atomic unit of the security policy of PA as shown in Figure 4.6. Therefore, $\iota(t)$ does not oscillate as $\iota_{ATE}(t)$ does and the value of $\iota(t)$ varies more than the value of $\iota_{ATE}(t)$. This scenario shows that the proposed model reflects changes in the security system of a

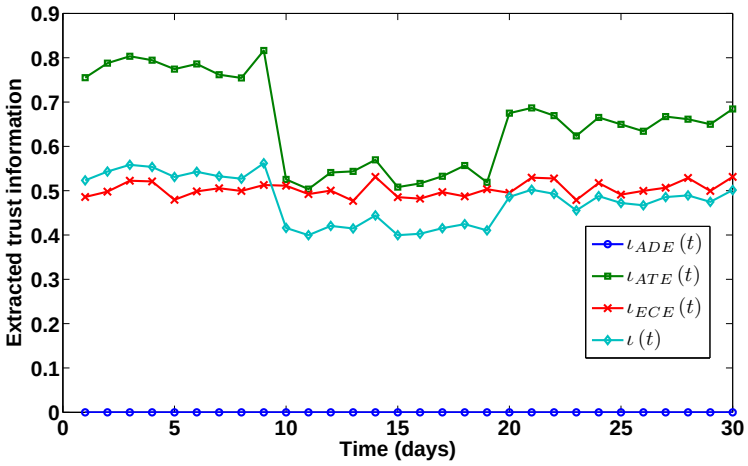

Figure 4.6. Extracted trust information when the security system of BDENT is changed.

service according to needs of an entity.

4.5.3. Scenario 2: Effects of Changes in Security Policies

Our goal in this scenario is to show effects of changes in the security policy of a service and in the security policy of an entity. Moreover, we present effects of overall history. Specifically, we update the security policy of BDENT and the security policy of entity PA by considering the facts in the previous scenario to accomplish the goal.

The monitoring software is turned on without updating the security policy of BDENT in Scenario 1. The security policy of BDENT is updated when the monitoring software is turned on at $t = 10$. A new rule related to the change is added to the security policy of BDENT as the third rule. The new rule is *Accesses to patients' records by dentists are logged by the monitoring software*, which is represented with atomic unit ADS. PA updates set $P^s_{BDENT}(t)$ at $t = 10$, where $P^s_{BDENT}(t) = \{ECS, ATS, ADS\}$. Additionally, perception factor $\pi_{ADE,ADS}(t)$ is changed to be 0.6 after $t = 10$.

On the other hand, managers of BDENT believe that storing patients' records in an encrypted form brings additional cost to manage the database of BDENT. They also believe that an adversary cannot access to the database of BDENT from the Internet

or from the physical environment. Therefore, the first rule of the security policy of BDENT is removed at $t = 25$. Additionally, PA updates set $P^s_{BDENT}(t)$ by removing atomic unit ECS from set $P^s_{BDENT}(t)$ at $t = 25$. The entity also changes perception factor to be $\pi_{ECE,ECS}(t) = 0$ for $t \geq 25$.

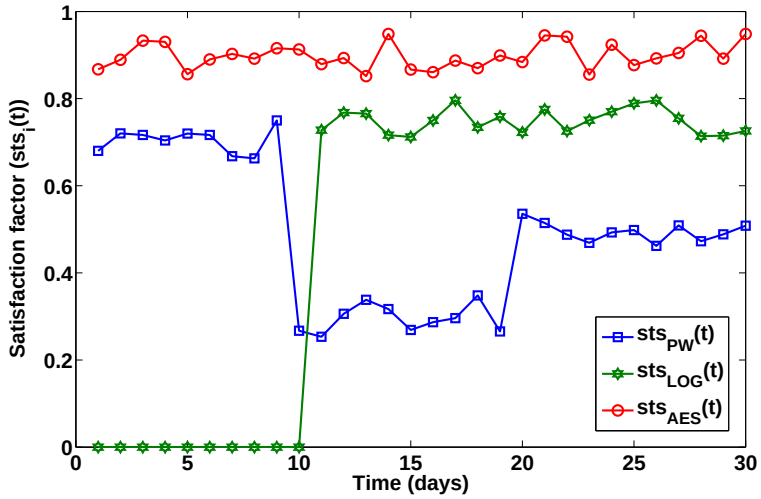

Figure 4.7. Satisfaction factors when the security policy of BDENT is changed.

Figure 4.7 shows changes in satisfaction factors after the security policy updates and Figure 4.8 shows the changes of extracted trust information related to the security policy change. Extracted trust information related to a rule of the security policy of BDENT is better reflected when there is an update in the rule as shown in these figures. Additionally, extracted trust information related to all atomic units behaves as expected. For instance, the value of $\iota(t)$ depends on extracted trust information related to each individual atomic unit as shown in Figure 4.8.

The owner of PA observes that many of patients' record services of dental clinics do not contain monitoring mechanisms. Moreover, the dental clinics do not share their logging data related to accesses to their security systems with entities. Therefore, the patient updates the security policy of PA by removing the third rule from the security policy at $t = 15$.

PA updates its set representations of security policies and set representation of the security system of BDENT according to the change of its security policy. In the

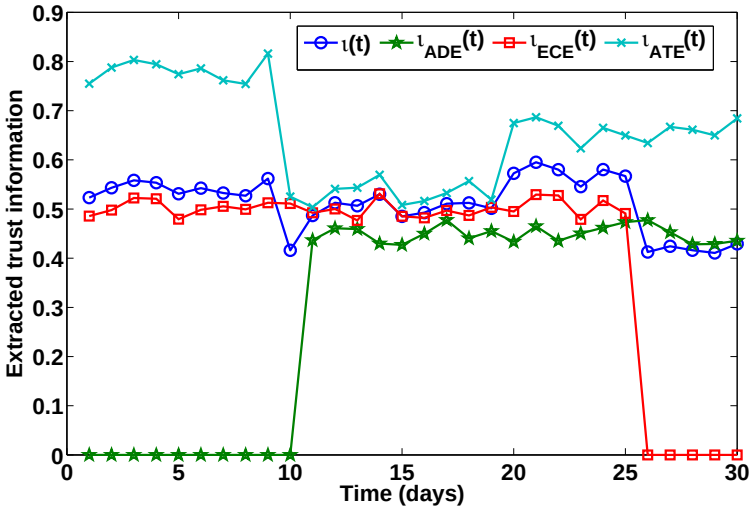

Figure 4.8. Extracted trust information when the security policy of BDENT is changed.

updated form, set $P^{e}_{BDENT}(t) = \{ECE, ATE\}$ after $t = 15$. Because atomic unit ADE is removed from set $P^{e}_{BDENT}(t)$, sets $P^{s}_{BDENT}(t)$ and $\Phi_{BDENT}(t)$ do not contain atomic units associated with atomic unit ADE after $t = 15$. Therefore, set representations are $P^{s}_{BDENT}(t) = \{ECS, ATS\}$ and $\Phi_{BDENT}(t) = \{AES, PW\}$ for $15 < t \leq 25$. Expected set representations are $P^{xs}_{BDENT}(t) = \{ECS, ATS\}$ and $\Phi^{x}_{BDENT}(t) = \{TPM, AES, PW\}$ for $t > 15$. Since the first rule is removed from the security system of BDENT at $t = 25$, the set representation become $P^{s}_{BDENT}(t) = \{ATS\}$ for $t > 25$.

Because of the change in the number of atomic units of set $P^{e}_{BDENT}(t)$, PA updates importance factors at $t = 15$. Specifically, we assume that the second rule of the security policy of the entity is more significant than the first rule. Therefore, $imp_{ECE} = 0.35$ and $imp_{ATE} = 0.65$ for $t > 25$ in this scenario.

Effects of changes in the security policy of PA is shown in Figure 4.9. The entity removes atomics units from its sets representing security policies and the security system that are related to the third rule of its security policy at $t = 15$. Therefore, $\iota(t)$ depends only on $\iota_{ECE}(t)$ and $\iota_{ATE}(t)$ after $t > 15$. Specifically, the value of $\iota(t)$ slightly increases because the value of $\iota_{ADE}(t)$ is always smaller than values of $\iota_{ECE}(t)$

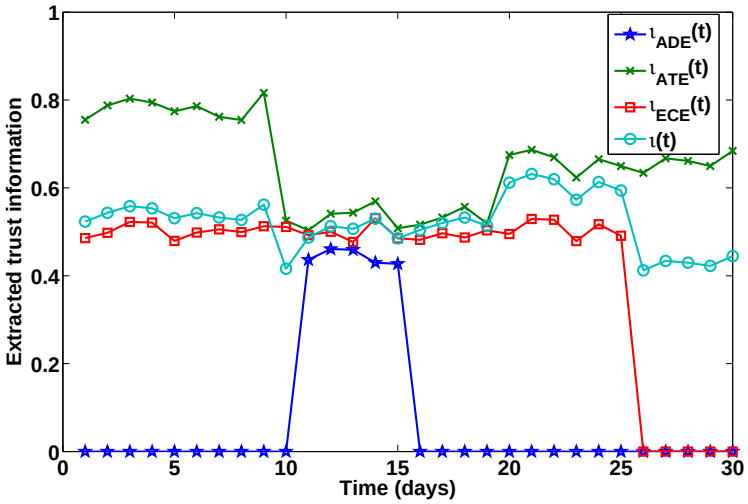

Figure 4.9. Extracted trust information after removing the first rule of the security policy of PA.

and $\iota_{ATE}(t)$ in this scenario. In short, our model reflects changes in the security policy of an entity to extract trust information related to the security of a service.

The experimental results that are shown in Figure 4.5-9 do not contain effects of histories. However, extracted trust information usually depends on the histories of the security system of a service and the security policy of the service. The overall history $h_j(t)$ related to atomic unit $p_j \in P^{xs}_{service}(t)$ is determined according to the history of the security policy $h^{ssp}_j(t)$ and the history of the security system of the service $h^{sss}_j(t)$. The effects of $h^{ssp}_j(t)$ and $h^{sss}_j(t)$ to $h_j(t)$ for all possible values of $h^{ssp}_j(t)$ and $h^{sss}_j(t)$ are shown in Figure 4.10. Additionally, trust information $ta_j(t)$ related to atomic unit $p_j \in P^{xs}_{BDENT}(t)$ is computed according to overall history $h_j(t)$ and the satisfaction factor $stp_j(t)$. Possible effects of $h_j(t)$ and $stp_j(t)$ to $ta_j(t)$ are shown in Figure 4.11.

Trust information of atomic unit $p_j \in P^{xs}_{service}(t)$ contributes to the computation of extracted trust information related to atomic unit $p_i \in P^{e}_{service}(t)$. On the other hand, each entity can evaluate histories of security policies of services and their corresponding histories of security system depending on their needs. Therefore, one can apply results in Figure 4.11 to the results in this case study to see possible effects of histories.

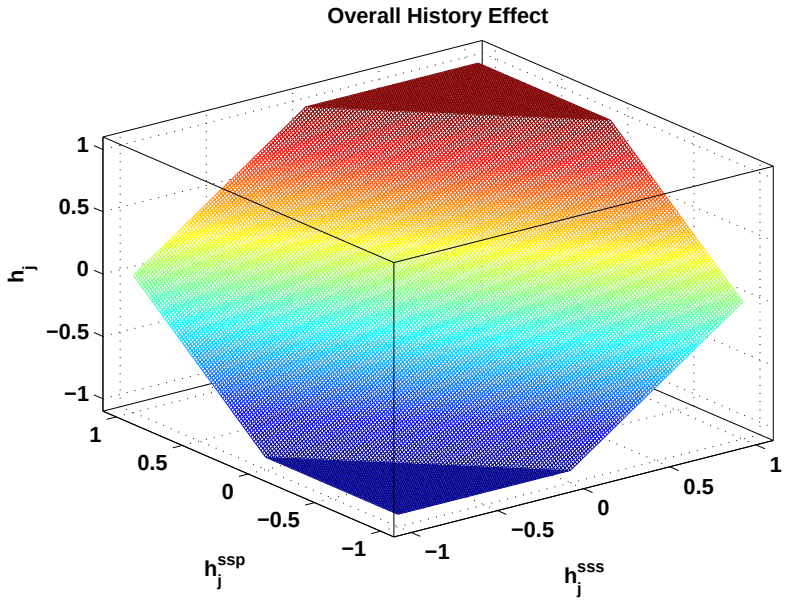

Figure 4.10. The change of overall history of atomic unit p_j in security policy of the service.

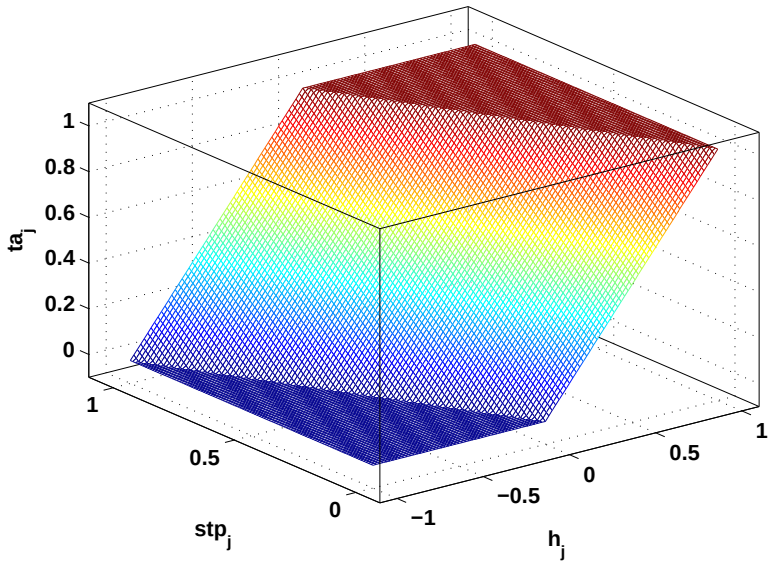

Figure 4.11. The change of trust information of atomic unit p_j related to overall history and the satisfaction factor.

This scenario shows that our model reflects changes in the security policy of an entity and the security policy of a service for extracting trust information related to the security system of the service based on needs of the entity. Moreover, the scenario presents possible history effects.

The case study shows that the proposed model is applicable to entities in open environments. Each entity can extract trust information related to the security system of a service by considering its security policy and the security policy of the service even though the security policies and the security system may be dynamic. Moreover, an entity can evaluate the security system of a service based on its present needs.

4.6. Conclusion

Open environments are going to support a large number of various services that interact with many different autonomous entities. Such diversity of services leads to trust problems in entities related to security systems of services. Moreover, the trust problems create new research challenges in emerging open environments. One such challenge is to obtain information related to the security system of a service for trust computations. In this chapter, we studied the challenge of obtaining trust information from the security system of a service in emerging open environments.

We proposed a crust model for extracting trust information from the security system of a service based on needs of a specific entity in emerging open environments. The security needs of an entity from a service are represented in the security policy of the entity. The security system of a service is an enforcement of the security policy of the service. Therefore, we have considered the security policy of an entity, the security policy of a service and the security system of the service for extracting trust information.

In the proposed model, security policies and security systems are represented with sets of atomic units. A security policy consists of rules, where each rule is an atomic unit. On the other hand, a security system consists of security mechanisms that are

represented with atomic units. Therefore, an entity has a set for atomic units of its security policy, a set for atomic units of the security policy of a service, and a set for atomic units of the security system of the service. An entity only represents rules of security policies that are related to needs of the entity from the service. Furthermore, the entity represents only security mechanisms of the service that are related to needs of the entity.

In our model, an entity can extract trust information about a specific atomic unit and trust information about all atomic units. Specifically, an entity can extract trust information related to a specific rule of its security policy that means the entity may need to determine the trust of a specific security property of the security system of a service. The entity may also need to have trust information related to whole security system of the service so it can extract trust information about whole security system.

A case study for management service of patients' accounts of dental clinic BDENT was presented with two scenarios to show the way to extract trust information from the security system of a service. We simulated the scenarios to evaluate the proposed model. In the first scenario, we analyzed effects of changes in the security system of the service. In the other scenario, we investigated effects of changes in the security policy of an entity and changes in the security policy of the service. The evaluation results show that the proposed model can provide information about the security system of a service based on specific needs of an entity for trust computations in emerging open environments.

5. INFORMATION FLOW CRUST MODEL: A MODEL FOR TRUST ASSESSMENT BASED ON FLOW OF SECURITY EVALUATION INFORMATION ON ENTITIES

Emerging networks are expected to be service-oriented environments that are highly dynamic. Entities in such environments have different security needs from security systems of services. A security system is a set of security mechanisms. Management of security evaluation information in dynamic environments with multiple entities, each with its own changing needs, is a complex task. The complexity mainly arises from the lack of trust to security evaluation information collected from entities and services. Therefore, the trust assessment of the security system of a service depends on the propagation of security evaluation information in the network. In this chapter, we present a crust model for flow of security evaluation information on entities and a model for trust assessment based on the security evaluation information. The proposed security evaluation information flow model and the trust assessment model have been applied to an online hotel reservation service as a case study. Two proposed models have been evaluated experimentally with simulations in the case study. The case study and the experimental evaluation result in more accurate trust assessments of security systems of services in emerging networks.

The rest of the chapter is organized as follows: We present our motivation in Section 5.1. We discus related work in Section 5.2. Section 5.3 describes the model for flow of security evaluation information on entities. We present our trust assessment model about the security system of a service in Section 5.4. We present an online hotel reservation service case study with an experimental evaluation in Section 5.5. Finally, we conclude in Section 5.6.

5.1. Introduction

Emerging networks are expected to be heterogeneous networks that converge to a ubiquitous network. The ubiquitous network is an open environment that supports large number of various services. Traditional network services are specific to networks. On the other hand, services in convergent networks must be accessible without considering the underlying access network. Service convergence necessitates managing services over a common environment [114]. Therefore, service-oriented computing is an emerging aspect of computer science and technology.

A service-oriented environment contains services and autonomous entities. The web is an instance of service-oriented environment. World Wide Web Consortium (W3C) defines a web service as a software system designed to support interoperable machine to machine interaction over a network. A web service is a set of related functionalities that can be manipulated over the web [110].

The diversity of services increases in computer networks and trust problems related to security issues become apparent. Computer security includes topics such as protection of information from theft and corruption that are discussed in the categories of authentication, confidentiality, integrity, and availability [2, 18]. On the other hand, trust has been investigated in various fields of science, such as philosophy [42], psychology [43], sociology [44] and computer science [1, 39, 57, 109], but there is no consensus about the definition of trust that meets all requirements in all research fields. The same situation is also true for properties of trust.

There are two main components in trust computation systems, the source of trust information and the consumer of trust information. Trust information generally flows from a source to a destination according to a trust propagation model [120]. An experience is trust information that flows from the source to the destination without passing over another party. If information flows from the source over another component or components to the destination, such information is called a recommendation. In computer networks, an entity obtains experiences and receives recommendations for

computations of trust about services, such as peer-to-peer reputation management [86].

Trust assessment of the security system of a service depends on information gathered from different sources, where trust assessment is a kind of trust computation. Security evaluation information can be any sensitive information related to the security system of a service, such as properties of authentication credentials and cryptographic algorithms. Trust assessment is carried out by using security evaluation information related to the security system of a specific service. In online systems, an entity can obtain information related to the security system of a service directly from the service or indirectly from other entities. Indirect information has to be propagated from a source to a destination over entities. Therefore, security evaluation information that flows on entities necessitates a precise model of an entity.

5.1.1. Motivation

It is expected that emerging networks will enable one person to connect many networks to obtain services. The networks should be interconnected to ensure the access to diverse number of services. Services should be accessible without considering the underlying network and user hardware. For example, a web service may be accessible with a cell phone or a desktop computer, through a 3G network or a local area network.

Several new security problems arise because of the multiplicity and the diversity of services. One such problem is the lack of trust to the security system of a service. To make a decision based on trust one needs to perform trust assessment of a system. Each trust computation model has different types of information to compute trust metrics. Generally, there are two types of information related to trust computations, namely direct information and indirect information.

A trust assessment model needs information about the system for which the trust assessment will be carried out. The amount of information affects the time needed to compute the level of trust. In existing trust computation models, an entity considers only its experiences about a service for trust computations. Additionally, some models

also use recommendations from other entities to compute trust, such as the multi-hop recommendation protocol for ad-hoc environments [87]. Existing models are too general and they do not reflect behaviors of each entity during flow of security evaluation information. However, it is significant for critical systems to consider the behavior of each entity to assess trust more precisely, such as security systems of applications and services in emerging networks. Therefore, models for flow of security evaluation information that reflect the behavior of each node on information flow network are needed. Additionally, trust assessment models that are based on information flow are necessary.

5.1.2. Contributions

In this chapter, we propose a crust model for flow of security evaluation information on entities for trust assessment about the security system of a service. In the proposed model, an entity obtains experiences directly from one specific service and from additional services that have similar security systems with that specific service. Additionally, the entity receives recommendations and confidences of recommendations about the security system of the service from other entities. Based on the proposed model, we propose a trust assessment model about the security system of a service. The proposed model for flow of security evaluation information and the trust assessment model have been applied to an online hotel reservation service as a case study. Two proposed models have been evaluated experimentally with simulations. The case study and the experimental evaluation result in more accurate trust assessment of the security system of a service in emerging networks. We can summarize the contribution of our work as follows.

- We introduce a formal model related to flow of security evaluation information on entities about the security system of a service. The model enables an entity to obtain security evaluation information from services and entities related to the security system of a service.

- We introduce subjective factors, such as similarity ratio, risk factor, accuracy

factor. The subjective factors depend on entities and services and they reflect the behavior of each individual entity during flow of security evaluation information.

- We propose a model for trust assessment based on flow of security evaluation information on entities. An entity can use the assessed trust to make decisions about the security system of a service by considering its private needs and its own facts.

5.2. Related Work

The application context usually determines the trust model and the propagation of trust information from a source to a destination. Different types of trust propagation models are explored in [121] that models can be implemented with subjective logic, but it is not possible to represent all properties of trust with a single model [121] because of different needs of entities.

In literature, trust is determined primarily in a source. Then trust is transmitted to a destination according to different models. For instance, Guha *et al.* propose a framework for propagation schemes of trust and distrust by considering different circumstances [122]. A more specific study about propagation of trust and distrust is accomplished by Zhu *et al.* by considering co-citations [120]. Semantic based information trust computation and propagation algorithm for semantic web is proposed by Zhang *et al.* [123]. Lightweight distributed trust propagation is studied in [124]. Although these works contain many contributions to trust modeling, none of them shows a way to propagate security evaluation information for computing the trust of a service only in the final node by taking into account the subjective behavior of each entity.

In our model, differently from models mentioned above, first, security evaluation information about a service is transmitted over entities, then trust metrics are computed only in the last node. The last node can be an entity or any other system that is able to compute trust metrics. Furthermore, differently from existing works, our model for flow of security evaluation information better reflects subjective properties of en-

tities so that our trust assessment model provides more accurate assessment results about the security system of a service.

5.3. Security Evaluation Information

In this model, the trust assessment process is carried out on entities. Therefore, an entity has to obtain security evaluation information about the security system of a service. The security evaluation information flows from a service to an entity, where the trust assessment is accomplished. Specifically, an entity may obtain security evaluation information directly from a service or the entity may receive security evaluation information about the security system of the service from other entities indirectly. In this section, we explain our proposed model for flow of security evaluation information. The model is a formal representation of an entity's belief about the security system of a service.

5.3.1. Experience

If an entity has previously interacted with services, the entity has experiences about security systems of these services. Experiences are information obtained by direct interactions of an entity with services.

An entity may interact with more than one service at the same time, but the entity can evaluate security systems of services separately. The security system of a service may contain similar or the same ingredients with security systems of other services. Therefore, an entity may use experiences obtained from one service to compute the ultimate experience about another service. For example, suppose that an entity needs to use experiences obtained from service ω_j to compute the ultimate experience related to service ω_i. Assume that security systems of services ω_j and ω_i consist of password based authentication mechanisms. The security system of service ω_j accepts passwords that have at least six characters, but it does not distinguish between capital letters and small letters. On the other hand, the security system of service ω_i necessitates passwords to contain both capital letters and small letters, but it does not have any

constraint related to the length of a password. Experiences obtained from service ω_j may help to improve the amount of experiences related to the security system of service ω_i according to the similarity between security systems of the two services.

Although security systems of services may be similar, the security systems are not necessarily the same. Therefore, we define a similarity ratio, $\lambda \in [0, 1]$, which represents similarities between security systems of two services. Each entity may have different similarity ratios related to the same services. Similarity ratios in an entity depend on a service for which experiences are computed. Therefore, similarity ratios are determined by considering a specific entity and the security system of a service. Λ_i is the set of similarity ratios related to the security system of service ω_i, where $\omega_i \in \Omega$. Subjective factors like the similarity ratio may be computed as in [125]. The computation of a similarity ratio is presented in Chapter 6.

In this model, a similarity ratio is asymmetric. For instance, the similarity ratio about the security system of service ω_i related to the security system of service ω_j, λ_i^j, is not the same with the similarity ratio about the security system of service ω_j related to the security system of service ω_i, λ_j^i. Note that $\lambda_q^i > \lambda_y^i$, where $\lambda_q^i, \lambda_y^i \in \Lambda_i$, means the security system of service ω_q is more similar than the security system of service ω_y to the security system of service ω_i. In addition, zero means no similarity between two security systems and one means completely identical security systems.

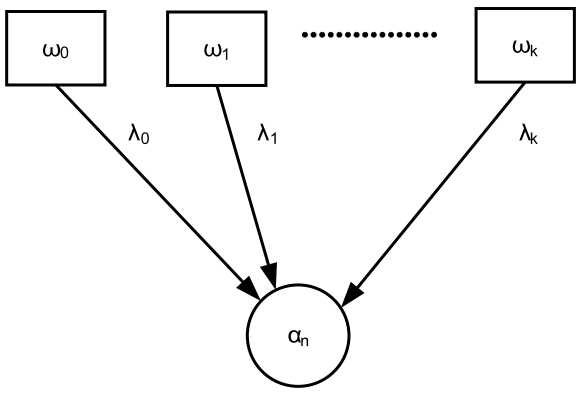

Figure 5.1. Directed acyclic graph for experiences of an entity.

5.3.1.1. Obtained Experience. An entity can obtain security evaluation information from services continuously so that experiences are time dependent. For this reason, we denote the experience obtained from service ω_i with $\epsilon_i^r(t) \in [0,1]$. If an entity obtains security evaluation information about all properties of the security system of service ω_i and the properties satify the needs of the entity, then $\epsilon_i^r(t)$ equals to one. In contrast, if there is no security evaluation information about any property of the security system of service ω_i or existing properties do not satisfy the needs of the entity, then the value of $\epsilon_i^r(t)$ is zero. On the other hand, $0 < \epsilon_i^r(t) < 1$ means that the security system of service ω_i partially satisies the needs of the entity.

5.3.1.2. Aggregated Experience. We represent the set of services that have interacted with an entity to obtain experiences about the security system of service ω_i at a specific time t with $\Omega_i^{(t)}$, where $\Omega_i^{(t)} \subseteq \Omega$. Each entity can construct a directed acyclic graph to compute the aggregated experience about the security system of service ω_i as in Figure 5.1. The aggregated experience about the security system of service $\omega_i \in \Omega$ is computed according to members of set $\Omega_i^{(t)}$. The aggregated experience about the security system of service ω_i is weighted sum of obtained experiences that are similar to the security system of service ω_i. In this model, $\epsilon_i^a(t) \in [0,1]$ represents the aggregated experience about the security system of service ω_i at a given time t in an entity. The aggregated experience about the security system of service ω_i is computed with Equation 5.1, where $W_i(t)$ is the number of elements in set $\Omega_i^{(t)}$.

$$\epsilon_i^a(t) = \frac{1}{W_i(t)} \sum_{\forall \omega_j \in \Omega_i^{(t)}} \epsilon_j^r(t) \lambda_j^i, W_i(t) > 0. \tag{5.1}$$

5.3.1.3. Ultimate Experience. An entity continuously obtains security evaluation information from services about the security system of service ω_i so that entities have histories of aggregated experiences. We consider all experiences for the computation of an ultimate experience about the security system of service ω_i. Specifically, we use

old experiences to improve the amount of experiences related to a service. Each entity has an aging factor that determines the amount of contribution of old experiences for computing recent experience. The aging factor for experiences is a subjective factor specific to an entity and each entity may have a different aging factor.

The ultimate experience about the security system of a service is the security evaluation information that takes into account the history of experiences. Note that the ultimate experience is the security evaluation information that is used for assessing the trust of the securiy system of a service. The ultimate experience of an entity about the security system of service ω_i at a given time t is represented with $\epsilon_i(t) \in [0,1]$ and is computed as in Equation 5.2. Greater $\epsilon_i(t)$s imply more valuable ultimate experiences.

$$\epsilon_i(t) = min\left(1, \sum_{x=0}^{t-1} \tau^{x+1} \epsilon_i^a(t-x)\right). \tag{5.2}$$

We assume that the most recent experiences are more significant than others. For this reason, we decrease the significance of older experiences by using an aging factor specific to an entity. We represent the aging factor with $\tau \in [0,1]$. The significance of history is linked with the value of the aging factor. For instance, if the history is less important, the value of the aging factor is smaller. An aging factor close to one causes old experiences to be evaluated like recent experiences. Therefore, few old experiences with higher values lead an ultimate experience to reach to one, which means old experiences are significant as recent experiences. If an entity has high valued ultimate experience with a high aging factor and it continues to obtain experiences close to one, the new obtained experiences do not increase the value of the ultimate experience so the ultimate experience may be normalized according to the number of experiences used. On the other hand, if an entity has an aging factor close to zero, old experiences are not so important for the computation of the ultimate experience. Additionally, if the ultimate experience is normalized, recent experiences may lose their significance. Therefore, we do not normalize the ultimate experience when we consider

aging factor. Actually, we normalize experiences when we compute the aggregate experience in Equation 5.1.

In our model, time is discrete and increases by one for each new computation of experiences.

5.3.2. Recommendation

An entity may receive security evaluation information about the security system of a service from other entities that we call a recommendation. In our model, a recommendation is indirect security evaluation information used for the computation of the assessed trust of the security system of a service.

An entity may receive recommendations from some entities and may send its recommendations to other entities. In this model, interacted entities with a specific entity are separated into two sets in the entity. The first set consists of entities from which the entity receives recommendations at a given time t and the set is represented with $A_r^{(t)}$, where $A_r^{(t)} \subset A$. The second set consists of entities to which the entity sends its recommendations at a given time t and the set is represented with $A_s^{(t)}$, where $A_s^{(t)} \subset A$, $A_r^{(t)} \cap A_s^{(t)} = \emptyset$, and $A_r^{(t)} \cup A_s^{(t)} \subset A$. In addition, an entity constructs a directed acyclic graph with entities it interacts as shown in Figure 5.2 to separate these two sets.

An entity may send different recommendations to each entity by considering its utility relations with these entities. For example, entity α_n may send recommendations to entity α_p with full accuracy at the time t because their goals may be the same and they must collaborate fully, where $\alpha_p \in A_s^{(t)}$, $\alpha_n \notin A_r^{(t)}$, and $\alpha_n \notin A_s^{(t)}$. On the other hand, entity α_n may change exact values of recommendations that it sends to entity α_g, where $\alpha_g \in A_s^{(t)}$ because of some conflicts with its goal and the goal of entity α_g. Therefore, each received recommendation may be perceived differently by an entity related to the security system of a service.

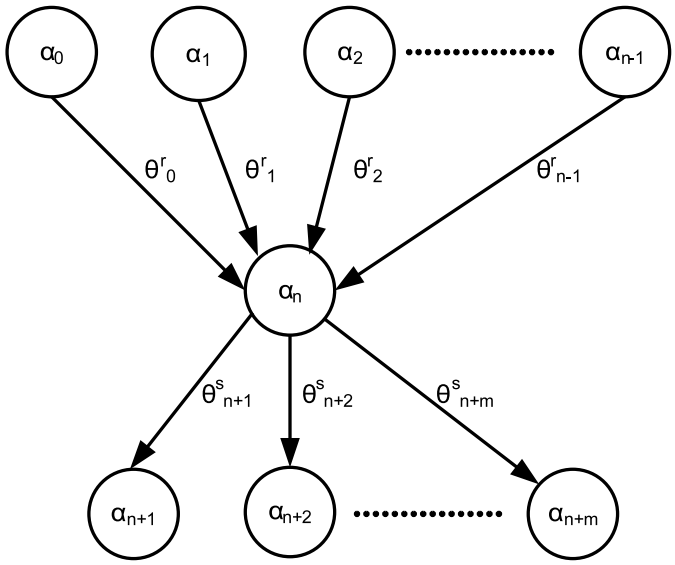

Figure 5.2. Directed acyclic graph for recommendations of an entity.

5.3.2.1. Accuracy Factor. We define an accuracy factor for a recommendation that is sent by entity α_n to entity α_g such that $\alpha_g \in A_s^{(t)}$. The accuracy factor for recommendations sent to entity α_g is represented with $\theta_g^s \in [-1, 1]$ in an entity. If accuracy factor θ_g^s is zero, entity α_n sends recommendations with full accuracy to entity α_g. Recommendations computed with $\left|\theta_g^s\right| = 1$ are completely inaccurate. Additionally, each entity may have different accuracy factors related to recommendations sent to a particular entity. The computation of an accuracy factor depends on interest relations between two entities and is out of scope of this thesis.

5.3.2.2. Risk Factor. Entities know that values of recommendations may be inaccurate so that false recommendations may mislead them. To cope with this problem, our entity model has a risk factor for each entity from which recommendations are received. The risk factor related to the recommendation received from entity α_d is represented with $\theta_d^r \in [0, 1]$, where $\alpha_d \in A_r^{(t)}$. Similar to accuracy factors, risk factors depend on the collaboration ratio between entities. Risk factor θ_d^r equals zero means maximum risk whereas one means no risk about recommendations received from entity α_d. Simply, the risk is inversely proportional with the value of risk factor θ_d^r. A risk factor is specific to an entity and each entity may have different risk computation models.

5.3.2.3. Received and Perceived Recommendations. A recommendation about the security system of service ω_i received from entity α_d at time t is represented with $\mu^r_{d,i}(t)$, where $\mu^r_{d,i}(t) \in [0,1]$. The value of a received recommendation is not the actual value that the receiver entity perceives. The perceived recommendation related to service ω_i received from entity α_d at the time t is represented with $\mu^p_{d,i}(t)$ and is computed as Equation 5.3, where $\mu^p_{d,i}(t) \in [0,1]$. Additionally, received confidence $\delta^r_{d,i}(t)$ is the confidence about the recommendation received from entity α_d about the security system of service ω_i at the time t. The computation of a received confidence is presented in Section 5.3.3.

$$\mu^p_{d,i}(t) = \begin{cases} 0 & ,\eta \geq \delta^r_{d,i}(t) \\ \mu^r_{d,i}(t)\,\delta^r_{d,i}(t)\,\theta^r_d & ,\eta < \delta^r_{d,i}(t) \end{cases} \tag{5.3}$$

Low confidence of a recommendation means that the received recommendation may not be accurate. Therefore, the entity which receives the recommendation should consider this fact. Each entity has an acceptance threshold for confidences of recommendations. The acceptance threshold is specific to an entity and all entities may have different acceptance thresholds for confidences of recommendations. An acceptance threshold is represented with $\eta \in [0,1]$. If received confidence $\delta^r_{d,i}(t)$ is smaller than acceptance threshold η, the entity set zero to perceived recommendation $\mu^p_{d,i}(t)$ that means the effect of received recommendation $\mu^r_{d,i}(t)$ to the value of perceived recommendation $\mu^p_{d,i}(t)$ is negative.

5.3.2.4. Aggregated Recommendation. An entity takes into account many recommendations received from different entities for the computation of the assessed trust. The aggregated recommendation related to the security system of a service is used for the aggregation of recommendations from many entities. The aggregated recommendation of an entity about the security system of service ω_i at a given time t is represented with $\mu^a_i(t)$ and is computed with Equation 5.4, where $\mu^a_i(t) \in [0,1]$ and $R(t) = \left|A^{(t)}_r\right|$.

$$\mu_i^a(t) = \frac{1}{R(t)} \sum_{\forall \alpha_d \in A_r^{(t)}} \mu_{d,i}^p(t), R(t) > 0. \tag{5.4}$$

5.3.2.5. Total Recommendation. Similar to experiences, the most recent recommendations about the security system of a service are more significant than former ones so we use an aging factor for recommendations that is represented with γ, where $\gamma \in [0, 1]$. Therefore, the total perceived recommendation about the security system of service ω_i at a given time t is represented with $\mu_i^T(t)$ and is computed as Equation 5.5, where $\mu_i^T(t) \in [0, 1]$. Note that aging factor for recommendations γ has the same properties with aging factor for experinces.

$$\mu_i^T(t) = min\left(1, \sum_{v=0}^{t-1} \gamma^{v+1} \mu_i^a(t-v)\right). \tag{5.5}$$

5.3.2.6. Importance Factor. In this model, we assume that experiences are more valuable than recommendations for the computation of ultimate recommendations. Importance factor k in Equation 5.6 shows this property, where $k \geq 1$ and $k \in \mathbb{R}$. Each entity may have different importance factors, where importance factors are determined according to models specific to entities.

5.3.2.7. Ultimate Recommendation. Recommendations of an entity depend also on experiences of the entity. Therefore, the ultimate recommendation about the security system of a service in an entity is also related to experiences of the entity. The ultimate recommendation about the security system of service ω_i at the time t is represented with $\mu_i(t)$ and is computed as Equation 5.6, where $\mu_i(t) \in [0, 1]$ and k is the importance factor that shows the significance of experiences for the computation of the ultimate recommendation.

$$\mu_i(t) = \left(\frac{\mu_i^T(t) + k\epsilon_i(t)}{k+1} \right). \tag{5.6}$$

5.3.2.8. Sent Recommendation. Since an entity may send recommendations about the security system of service ω_i with different values to each entity at the same time, the entity must decide which values of recommendation it should send to each entity. An entity sends a recommendation about the security system of service ω_i to entity α_p at a given time t by using sent recommendation that is represented with $\mu_{p,i}^s(t)$ and the sent recommendation is computed as Equation 5.7, where $\mu_{p,i}^s(t) \in [0,1]$ and $\alpha_p \in A_s^{(t)}$.

$$\mu_{p,i}^s(t) = \begin{cases} \min\left(1, \mu_i(t) + \theta_p^s\right) & , \theta_p^s \geq 0 \\ \max\left(0, \mu_i(t) + \theta_p^s\right) & , \theta_p^s < 0 \end{cases} \tag{5.7}$$

An entity computes sent recommendations according to accuracy factors as shown in Equation 5.7. For each type of recommendation, zero means no recommendation whereas one means maximum recommendation.

5.3.3. Confidence

The multiplicity of recommendation sources about the security system of a service brings into existence confidence problems about received recommendations in an entity. The confidence of a recommendation is the quality measure of the recommendation that an entity sends to other entities. When an entity sends its recommendations, the entity also gives confidences about these recommendations to other entities.

5.3.3.1. Reliability Factor. Entities can evaluate experiences and recommendations differently for the compution of confidences about the security system of a service.

We represent the difference with a reliability factor. The reliability factor of an entity about the security system of service ω_i is represented with σ_i, where $\sigma_i \in [0,1]$. If the security system of a service is more reliable from the entity point of view, the reliability factor of the service is greater.

5.3.3.2. Received and Sent Confidences. Since an entity can receive and can send confidences of recommendations, there are two types of confidences about recommendations, namely received confidence of a recommendation and sent confidence of a recommendation. The received confidence of a recommendation from entity α_d about the security system of service ω_i at a time t is represented with $\delta^r_{d,i}(t)$, where $\delta^r_{d,i}(t) \in [0,1]$ and $\alpha_d \in A_r^{(t)}$. The confidence that is sent to any entity about the security system of service ω_i at a given time t is represented with $\delta_i(t)$, where $\delta_i(t) \in [0,1]$.

In our model, ultimate experiences and ultimate recommendations are time dependent. Therefore, we take into account previous instances of ultimate experiences and ultimate recommendations to compute confidences of recommendations. However, each entity may not have the same resources for the computation of the confidence of a recommendation. For instance, an entity may have energy constraints so that the entity may consider only limited number of previous ultimate experiences and ultimate recommendations for the computation of the confidence of a recommendation. For this reason, we use only limited number of the most recent ultimate experiences and the most recent ultimate recommendations to compute the confidence of a recommendation. N denotes the most recent n instances of ultimate experiences and ultimate recommendations that contribute to the computation of the confidence of a recommendation.

Each entity can have a different reliability factor for each service so reliability factors are subjective and depend on entities. Moreover, entities may determine their reliability factors with models specific to them. For example, one entity may have high reliability to security systems that are made by some specific vendors whereas another may have a low reliability factor to such security systems. Therefore, we do not present

any model to determine a reliability factor in this thesis.

$$\delta_i(t) = \frac{\sigma_i}{N} \sum_{u=0}^{N-1} \left| \frac{\epsilon_i(t-u) + \mu_i(t-u)}{2} - |\epsilon_i(t-u) - \mu_i(t-u)| \right|. \tag{5.8}$$

Experiences and recommendations contribute positively to the computation of the confidences about recommendations while differences between ultimate experiences and ultimate recommendations decrease confidences. Sent confidence of a recommendation about the security system of service ω_i at a given time t is computed as Equation 5.8, where σ_i denotes the reliability factor of service ω_i on the entity. A recommendation with full confidence has a value equal to one whereas a recommendation with no confidence has a value equal to zero.

5.4. Trust Assessment

We define *trust* as the security expectation of an entity from a service according to available security evaluation information of that entity. Trust is represented with trust metrics. A trust metric is computed with algorithms, which depend on an entity's utility. In our model, we have a trust metric for assessing the trust of the security system of a service, namely assessed trust. A trust metric is computed only in the destination node. The destination node represents an entity, where the trust assessment is carried out in our model.

Since we define trust as the security expectation of an entity about the security system of a service according to available security evaluation information on that entity, the trust is assessed by considering the available security evaluation information. Experiences, recommendations, and confidences of recommendations are security evaluation information. Therefore, an entity does the trust assessment by using experiences, recommendations and confidences of recommendations.

In our model, entities are autonomous so the trust computation model of an entity depends on the utility function of the entity. Each entity may have different trust computation approach because of the autonomous behavior of entities. Moreover, entities may have many trust metrics related to the security system of a service.

In this chapter, the trust of the security system of a service is assessed with a single metric. The *assessed trust* in an entity related to the security system of service ω_i at a given time t is the output of the utility function of the entity related to trust assessment as shown with Equation 5.9. Additionally, the trust assessment metric denotes the trustworthiness of the security system of a service from the entity point of view. We represent the assessed trust of the security system of service ω_i at a given time t with $t_i^a(t)$, where $t_i^a(t) \in [0,1]$.

$$t_i^a(t) = u^{ta}\left(\epsilon_i(t), \mu_i(t), \delta_i(t)\right)(t). \tag{5.9}$$

The value of the assessed trust shows the trustworthiness of a security system. If the value of the assessed trust is high, the security system is more trustworthy. For instance, if $t_i^a(t) = 1$, the security system of service ω_i has maximum trust on the entity. Whereas, if $t_i^a(t) = 0$, the security system of service ω_i has minimum trust or no trust on the entity.

Trust is a combination of direct and indirect information so the trust assessment metric may be computed with various formulas. For instance, the assessed trust may be computed with Equation 5.10. In this case, if an entity has no confidence to recommendations related to service ω_i that means $\delta_i(t) = 0$, then $t_i^a(t) = \epsilon_i(t)$ that is not realistic. Therefore, Equation 5.10 may not provide accurate results. A trust assessment model should consider both direct information and indirect information in any case.

$$t_i^a(t) = \frac{\epsilon_i(t) + \mu_i(t)\,\delta_i(t)}{1 + \delta_i(t)}. \tag{5.10}$$

All entities have the same utility function for the computation of the assessed trust in this chapter, which means that all entities compute the assessed trust metric in the same way. Specifically, the assessed trust is the mean of direct and indirect security evaluation information as shown with Equation 5.11. Moreover, the meaning of the trust assessment metric is that an entity can trust to the security system of a service if it has both direct security evaluation information and indirect security evaluation information related to the security system of the service.

$$t_i^a(t) = \frac{\epsilon_i(t) + \mu_i(t)\,\delta_i(t)}{2}. \tag{5.11}$$

The outcome of Equation 5.11 depends on both direct information and indirect information. For example, if an entity has either $\mu_i(t) = 0$ or $\delta_i(t)$, then $t_i^a(t) = \frac{\epsilon_i(t)}{2}$. In this case, the entity can never have a trust assessment result over 0.5 that means the entity does not have useful indirect information to verify its experiences. On the other hand, if the entity has $\epsilon_i(t) = 0$, then $t_i^a(t) = \frac{\mu_i(t)\delta_i(t)}{2}$. In the second case, the entity also will not have $t_i^a(t) > 0.5$ because it will not have useful direct information to verify indirect information. If the entity has both direct information and indirect information, such as $\epsilon_i(t) = 0.25$, $\mu_i(t) = 0.75$ and $\delta_i(t) = 0.15$, then $t_i^a(t) = 0.1813$. Moreover, if $\delta_i(t) = 1$ in previous example, $t_i^a(t) = 0.5$ that case shows the significance of a parameter in Equation 5.11 and correctness of the Equation.

In the proposed model, entities are in an open environment and they are able to interact with other entities and services. An entity obtains security evaluation information from services and entities after many interactions with them. We propose

Require: $\left|\Omega_i^{(t)}\right| > 0$ or $\left|A_r^{(t)}\right| > 0$, $N > 0$

Ensure: Trust assessment about the security system of service ω_i;

1: Obtain all $\epsilon_j^r(t)$ such that $\omega_j \in \Omega_i^{(t)}$;

2: Receive all $\mu_{p,i}^r(t)$ and all $\delta_{p,i}^r(t)$ such that $\alpha_p \in A_r^{(t)}$ and $\omega_i \in \Omega$;

{Compute the ultimate experince about service ω_i}

3: $\epsilon_i(t) \leftarrow \text{UExperience}(\Omega_i^{(t)}, \Lambda_i, \tau)$;

{Compute the ultimate recommendation about service ω_i}

4: $\mu_i(t) \leftarrow \text{URecommendation}(A_r^{(t)}, \epsilon_i(t), \delta_{p,i}^r(t), \gamma, k)$;

{Compute the confidence of recommendation about service ω_i}

5: $\delta_i(t) \leftarrow \text{Confidence}(\epsilon_i(t), \mu_i(t), N, \sigma_i)$;

6: $t_i^a(t) \leftarrow (\epsilon_i(t) + \mu_i(t)\,\delta_i(t))/2$;

7: **return** $t_i^a(t)$;

Figure 5.3. Trust Assessment Algorithm.

an algorithm as shown in Figure 5.4 for the computation of the assessed trust metric for entities, where the algorithm is designed according to the architectural approach in Chapter 3.

Briefly, any trust assessment approach has two main stages. The first stage is information gathering related to the trust assessment and the second stage is computations of trust metrics. The entity model for flow of security evaluation information is the first stage, where an entity gathers security evaluation information for trust assessments. The second stage is the trust assessment based on information gathered according to the entity model.

5.5. Case Study: Online Hotel Reservation Service

In this section, we analyze the proposed model by using an online hotel reservation service as a case study. The case study contains a scenario where an entity represents a travel agent. The agent collects security evaluation information about the security system of the online reservation service of a hotel. Next, the entity assesses the trust of the security system of the hotel. We simulate the scenario and show the performance

results of the proposed models.

Assume that the environment for online hotel reservation scenario contains three hotels, a travel agent, two businessmen, a politician, a professor, a student, and a tourist. In this case study, the hotels are services. The travel agent, the businessmen, the politician, the student, and the tourist are entities. The interactions among entities and services are shown in Figure 5.4. Moreover, the figure contains information flow among entities and services. For the sake of simplicity, we assume that the environment is static and the interactions in the environment do not change.

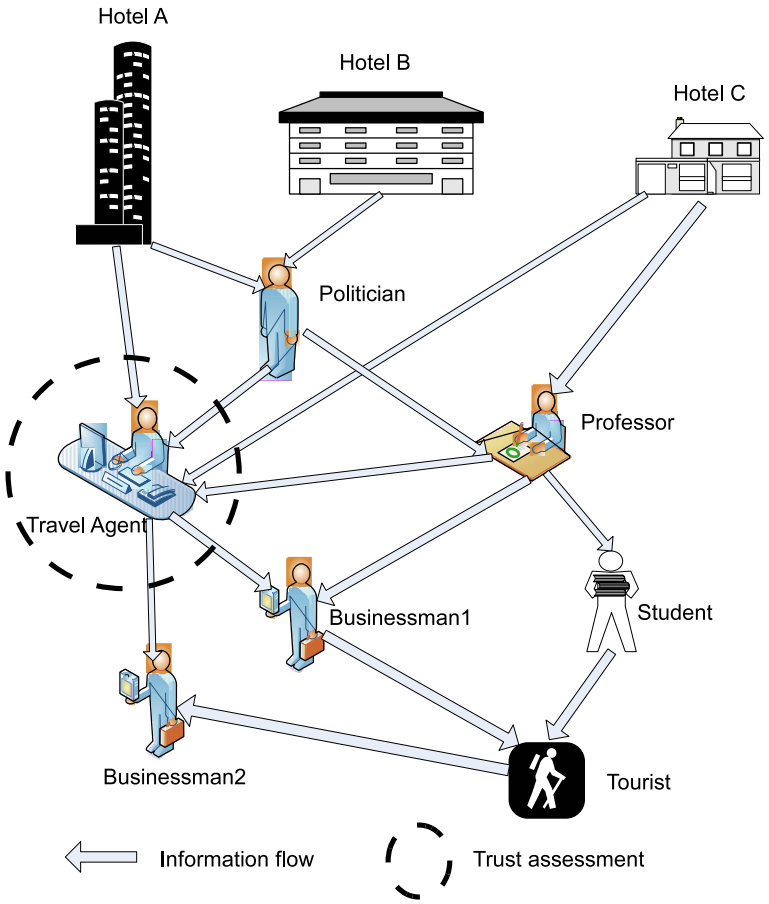

Figure 5.4. Information flow and interactions for online hotel reservation.

The travel agent can recommend hotels in a particular location to its customers. The customers of the travel agent need to trust to the security systems of online reservation services of hotels before performing further interactions for reservations. Furthermore, the customers should have trust to the recommendations of the travel agent. On the other hand, the travel agent recommends hotels that have a trust value

greater than a particular value. Therefore, the travel agent has to assesse trust of the security system of an online reservation service.

In our scenario, *Travel Agent* needs to assess the trust of the security system of the online registration service of *Hotel A*. The agent collects security evaluation information directly from hotels and indirectly from other people as shown in Figure 5.4. Then, *Travel Agent* assesses the trust of the security system of the online reservations service of *Hotel A*. Assume that *Politician* and *Businessman1* collaborate with *Travel Agent* so they have the same goals. In contrast, *Businessman2*, *Student*, and *Tourist* may collaborate partially because of several reasons.

5.5.1. Network Representation of Security Evaluation Information Flow

The environment in our scenario contains three services and seven entities. The set of services is $\{HotelA, HotelB, HotelC\}$, which corresponds to $\Omega = \{\omega_1, \omega_2, \omega_3\}$. The set that contains entities *Politician, Travel Agent, Professor, Student, Businessman1, Businessman2,* and *Tourist* corresponds to $A = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6, \alpha_7\}$. Additionally, the representation of the network for flow of security evaluation information related to the online hotel registration is shown in Figure 5.5. In this case study, trust assessments are carried out only in entity α_2. Therefore, we concentrate to compute security evaluation information in entity α_2.

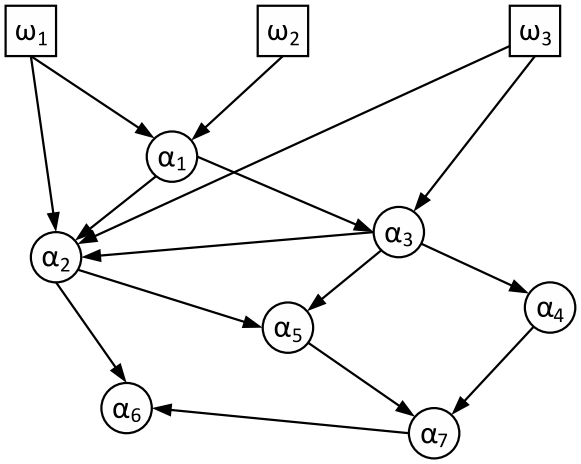

Figure 5.5. The network of security evaluation information flow.

The task of entity α_2, which is *Travel Agent*, is to collect security evaluation information about the security system of service ω_1, which service is the online registration service of *Hotel A*. Entities, α_1, α_2, and α_5, have identical tasks because of their goals so that entities α_1, α_2, and α_5 collaborate fully. On the other hand, entities $\alpha_3, \alpha_4, \alpha_6$, and α_7 may collaborate partially with entities α_1, α_2, and α_5.

Since an entity can interact with different services and different entities at different times, the network representation of security evaluation information flow is dynamic. Because we assume that the environment is static and the interactions in the environment do not change, the network related to flow of security evaluation information does not change in this scenario. For the sake of brevity, similarity ratios about the security system of service ω_1 in all entities are the same, $\lambda_1^1 = 1$, $\lambda_2^1 = 0.9$, and $\lambda_3^1 = 0.85$.

Each entity decreases significances of old experiences and old recommendations by using aging factors. Table 5.1 shows values of aging factors related to experiences and recommendations of all entities. Moreover, Table 5.1 shows η, k, and σ_1 of all entities.

Table 5.1. Some facts about entities.

	α_1	α_2	α_3	α_4	α_5	α_6	α_7
τ	0.25	0.69	0.82	0.7	0.9	0.79	0.88
γ	0.66	0.76	0.72	0.57	0.69	0.54	0.48
η	0.1	0.04	0.07	0.12	0.26	0.01	0.3
k	2.3	2.79	1.77	3.2	1.98	2.02	4.3
σ_1	0.93	0.81	0.87	0.97	0.98	0.99	0.78

Because we assume that the network representation of security evaluation information flow is static, set $\Omega_{HotelA}(t)$ is time independent for all entities. The interaction sets of all entities related to services are constructed according to the network representation in the environment as shown in Figure 5.5, where $\Omega_{HotelA} = \{\omega_1, \omega_2\}$ for entity

α_1, $\Omega_{HotelA} = \{\omega_1, \omega_3\}$ for entity α_2, and $\Omega_{HotelA} = \{\omega_3\}$ for entity α_3. Directed acyclic graphs for experiences of entities α_1, α_2, and α_3 are shown in Figure 5.6. Entities α_4, α_5, α_6, and α_7 have $\Omega_{HotelA} = \oslash$.

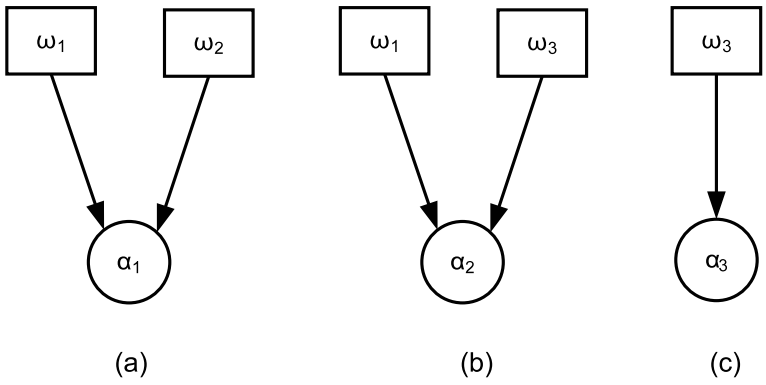

Figure 5.6. Directed acyclic graphs for experiences of entities (a) α_1 (b) α_2 (c) α_3.

Since we assume that the trust assessment is accomplished only in entity α_2, we explain only how security evaluation information flows from services to entity α_2. Entity α_2 obtains experiences from service ω_1 and service ω_3 as shown in Figure 5.6 and recommendations from entity α_1 and entity α_2 as shown in Figure 5.7.

An entity has accuracy factors for each element in its A_s. For example, $A_s = \{\alpha_2, \alpha_3\}$ for α_1. In this case study, accuracy factors of entity α_1 are $\theta_2^s = 0$ and $\theta_3^s = -0.08$ about entity α_2 and entity α_3 respectively. Accuracy factors of entity α_2 about entity α_5 and entity α_6 are $\theta_5^s = 0$ and $\theta_6^s = 0.3$ in sequence. $\theta_2^s = 0.1$, $\theta_4^s = -0.1$, and $\theta_5^s = 0.55$ are accuracy factors of entity α_3 about entities α_2, α_4, and α_5.

Similar to accuracy factors, an entity has risk factors related to received recommendations for each element in A_r. Entity α_1 does not receive any recommendation so that A_r is an empty set for entity α_1. On the other hand, $A_r = \{\alpha_1, \alpha_3\}$ for entity α_2 and risk factors of entity α_2 are $\theta_1^r = 0.95$ and $\theta_3^r = 0.62$ related to entity α_1 and entity α_3. Additionally, the risk factor of entity α_3 about entity α_1 is $\theta_1^r = 0.82$.

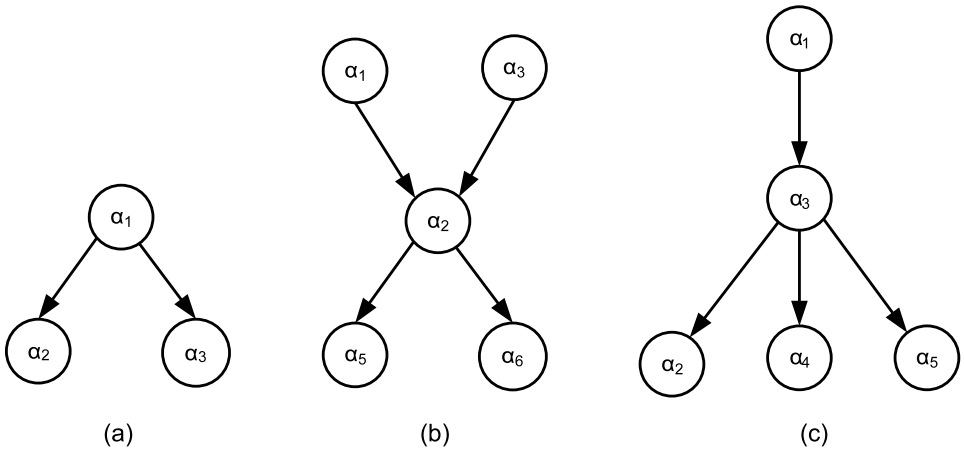

Figure 5.7. Directed acyclic graphs for recommendations of entities (a) α_1 (b) α_2 (c) α_3.

5.5.2. Experimental Evaluation

We illustrate the advantage of our proposed model for the online hotel reservation with simulations. We show the performance results about the computation of experiences, recommendations, and confidences of recommendations. Moreover, we simulate the trust assessment process at *Travel Agent* for the online hotel reservation to explain the trust assessment based on the proposed flow of security evaluation information on entities.

An entity obtains experiences, recommendations, and confidences of recommendations every second and completes computations within one second. Moreover, the trust assessment is carried out every second in an entity. Simulations were carried out by using MATLAB R2009b version 7.9.0.529.

5.5.2.1. Computation of Security Evaluation Information. We illustrate the flow of security evaluation information on entities by simulations. Specifically, we show the computation of the ultimate experience, the ultimate recommendation, and the confidence of the recommendation about the security system of the online registration service of *Hotel A*. Since the trust assessment about the security system of the online registration service of *Hotel A* is carried out on *Travel Agent*, the computation of ultimate experiences, ultimate recommendations, and confidences of recommendations are simulated

only for *Politician*, *Professor*, and *Travel Agent*.

Table 5.2. Aggregated experiences and ultimate experiences of entities α_1, α_2, and α_3.

t	$\epsilon_1^r(t)$	$\epsilon_2^r(t)$	$\epsilon_3^r(t)$	$\epsilon_1^a(t)$			$\epsilon_1(t)$		
				α_1	α_2	α_3	α_1	α_2	α_3
0	0	0	0	0	0	0	0	0	0
1	0.0605	0.3993	0.5269	0.2099	0.2542	0.4478	0.0525	0.1754	0.1721
2	0.4168	0.6569	0.6280	0.5040	0.4753	0.5338	0.1391	0.4490	0.5544
3	0.2920	0.4317	0.0155	0.3402	0.1526	0.0132	0.1198	0.4151	0.7336
4	0.9841	0.1672	0.1062	0.5673	0.5372	0.0903	0.1718	0.6570	1
5	0.3724	0.1981	0.4897	0.2754	0.3943	0.4162	0.1118	0.7254	1
6	0.3395	0.9516	0.9203	0.5980	0.5609	0.7823	0.1774	0.8876	1
7	0.0527	0.7379	0.2691	0.3584	0.1407	0.2288	0.1340	0.7095	1
8	0.4228	0.5479	0.9427	0.4580	0.6121	0.8013	0.1480	0.9119	1
9	0.4177	0.9831	0.3015	0.6512	0.3370	0.2562	0.1998	0.8617	1
10	0.7011	0.6663	0.5391	0.6504	0.5797	0.4583	0.2126	0.9946	1

In these simulations, all obtained experiences from hotels are uniformly distributed with means equal to 0.5. Moreover, we assume that entities can obtain experiences, recommendations, and confidences of recommendations every second.

Let $\epsilon_i^r(t)$'s be the same for all entities related to each web service at a given time. Table 5.2 shows the initial values of $\epsilon_1^a(t)$ and $\epsilon_1(t)$ depending on $\epsilon_1^r(t)$, $\epsilon_2^r(t)$, and $\epsilon_3^r(t)$ of entities α_1, α_2, and α_3.

Aggregated experiences depend on experiences obtained at a given time. However, previous obtained experiences affect values of ultimate experiences. Figure 5.8 shows the initial ten seconds of the computations of aggregated experiences and ultimate experiences about the security system of service ω_1. The aggregated experience depends on obtained experiences and similarity ratios. However, the ultimate expe-

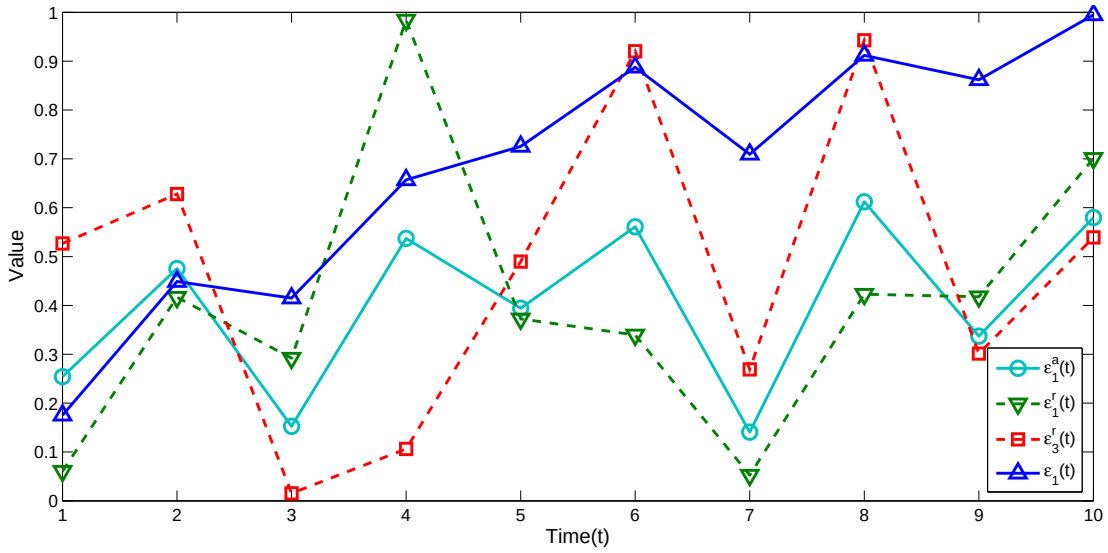

Figure 5.8. Changes in $\epsilon^a(t)$'s and $\epsilon(t)$'s of entity α_2 according to $\epsilon^r(t)$'s.

rience depends also on an aging factor so previous experiences are important for the computation of the ultimate experience about the security system of a service. Therefore, the variance of the ultimate experience is relatively small according to the variance of the aggregated experience.

The aging factor affects the computation of the ultimate experience. For instance, if the aging factor is small, then the ultimate experience never reaches to one, such as the ultimate experience about the security system of service ω_1 on entity α_1 in long run as shown in Figure 5.9. However, if an entity has a greater value of the aging factor related to experiences, the value of the ultimate experience can reach to one. For instance, ultimate experiences of entity α_2 and by entity α_3 computed in long run are shown in Figure 5.10 and Figure 5.11 respectively.

Actually, the aging factor may be related to the size of memory of an entity dedicated for storing experiences. Because the effects of older experiences on an entity depend on the aging factor, the entity may omit some older experiences. Therefore, the entity may use limited memory for computing an ultimate experience. If the entity runs on a resource limited platform, such as on a sensor, the resource consumption will become significant for the entity.

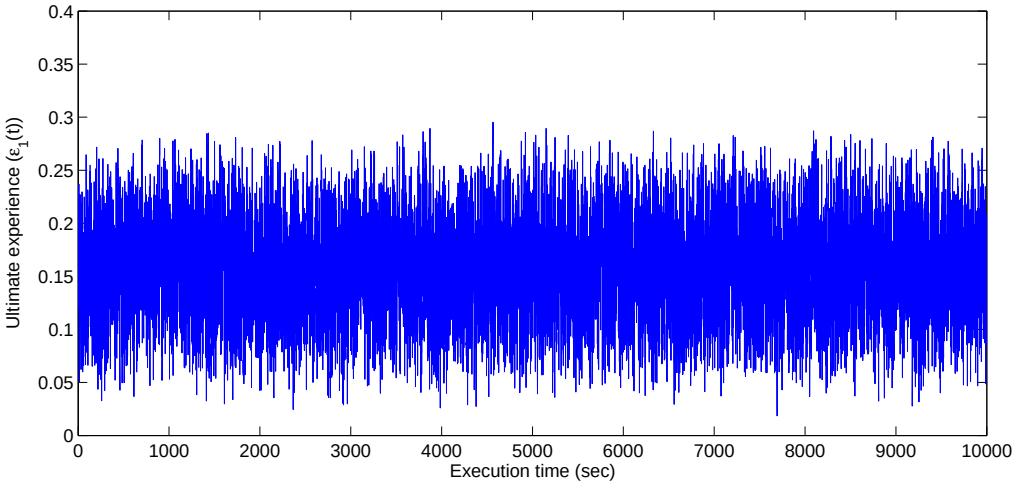

Figure 5.9. Ultimate experience $\epsilon_1(t)$ computed by entity α_1 in long run.

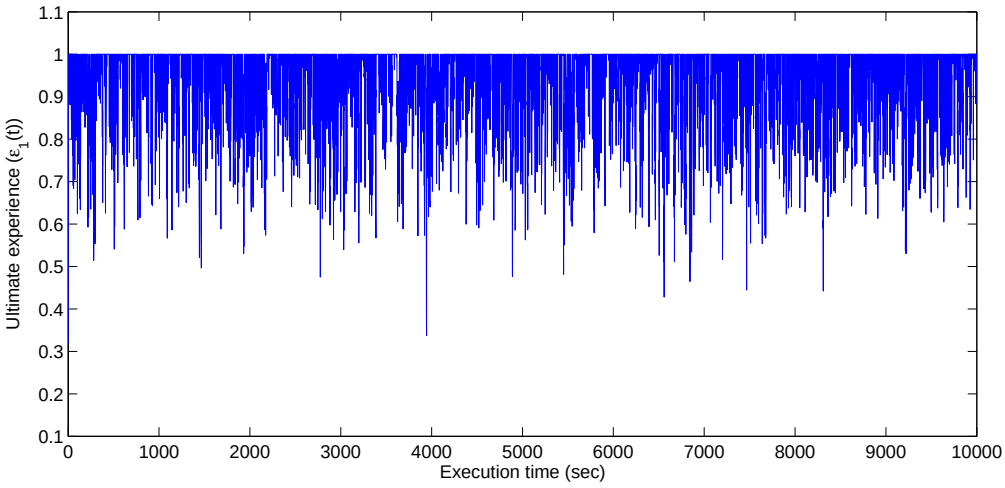

Figure 5.10. Ultimate experience $\epsilon_1(t)$ computed by entity α_2 in long run.

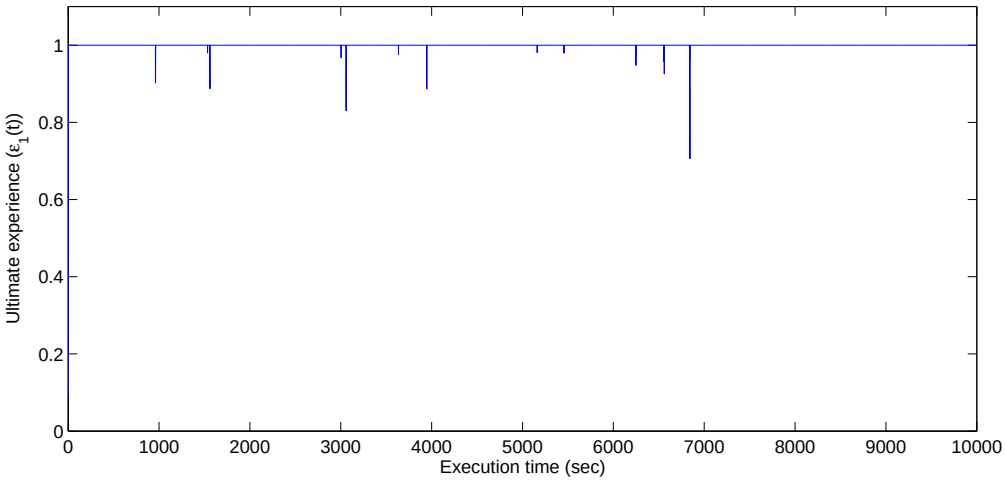

Figure 5.11. Ultimate experience $\epsilon_1(t)$ computed by entity α_3 in long run.

Table 5.3. $\mu_1^a(t)$, $\mu_1^T(t)$, and $\mu_1(t)$ computed by entities α_1, α_2, and α_3.

	$\mu_1^a(t)$			$\mu_1^T(t)$			$\mu_1(t)$		
t	α_1	α_2	α_3	α_1	α_2	α_3	α_1	α_2	α_3
0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0.0366	0.1291	0.1100
2	0	0	0	0	0	0	0.0970	0.3305	0.3543
3	0	0.0136	0	0	0.0103	0	0.0835	0.3083	0.4688
4	0	0.0363	0	0	0.0355	0	0.1197	0.4939	0.6390
5	0	0.0738	0.0024	0	0.0831	0.0017	0.0779	0.5559	0.6396
6	0	0.0859	0	0	0.3395	0.1284	0.1237	0.6873	0.6394
7	0	0.0961	0.0028	0	0.1707	0.0029	0.0934	0.5673	0.6400
8	0	0.0949	0.0008	0	0.2018	0.0026	0.1031	0.7245	0.6399
9	0	0.0955	0.0015	0	0.2260	0.0030	0.1393	0.6940	0.6401
10	0	0.0972	0.0040	0	0.2456	0.0050	0.1481	0.7970	0.6408

Ultimate recommendations depend on both experiences and recommendations. However, an entity may not have both experiences and recommendations at a given time. Therefore, ultimate recommendations may be computed either with experiences or with recommendations. In the case study, entity α_1 considers only experiences whereas entity α_5 takes into account only recommendations for the computation of the ultimate recommendation about the security system of service ω_1.

Table 5.3 shows initial steps of computations of $\mu_1^a(t)$, $\mu_1^T(t)$, and $\mu_1(t)$ on entities α_1, α_2, and α_3 about the security system of service ω_1. Since entity α_1 does not receive any recommendation, $\mu_1^a(t)$ and $\mu_1^T(t)$ are zero on entity α_1. However, the value of $\mu_1(t)$ computed by entity α_1 is not zero because entity α_1 obtains experiences from service ω_1 and service ω_2. In this case study, initial recommendations are zero in all entities. Therefore, if an ultimate recommendation is computed only by considering recommendations from other entities on an entity, the entity will never have an ultimate recommendation greater than zero.

The ultimate recommendation about the security system of a service is computed according to some factors in addition to the ultimate experience and obtained recommendations. Each factor affects the value of the ultimate recommendation. Therefore, each entity may have different ultimate recommendations related to the security system of a specific service. For instance, Figure 5.12 contains ultimate recommendation $\mu_1(t)$ computed by entities α_1, α_2, and α_3 in long run, where each entity has different values for ultimate recommendation $\mu_1(t)$ depending on facts of each entity.

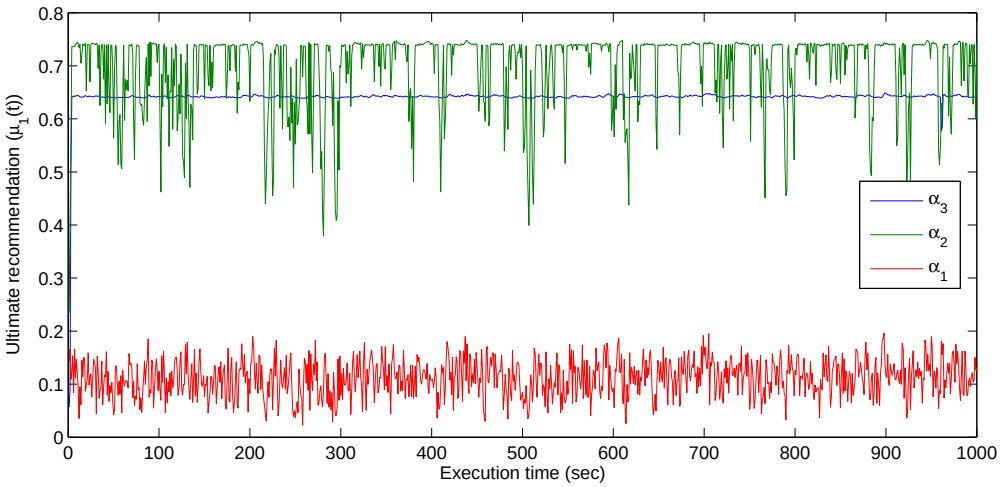

Figure 5.12. Ultimate recommendation $\mu_1(t)$ computed by entities α_1, α_2, and α_3 in long run.

The importance factor increases the contribution of experiences. The risk factor decreases the effects of obtained recommendations. Therefore, the value of an ultimate recommendation depends more on experiences than recommendations. For example, the behavior of ultimate recommendation $\mu_1(t)$ computed by entity α_2 as shown in Figure 5.12 are more similar to the behavior of ultimate experience $\epsilon_1(t)$ computed by entity α_2 as shown in Figure 5.10.

Table 5.4. $\mu^s(t)$s and $\delta_1(t)$s computed by entities α_1, α_2, and α_3.

t	α_1		α_2		α_3			$\delta_1(t)$		
	$\mu^s_{2,1}(t)$	$\mu^s_{3,1}(t)$	$\mu^s_{5,1}(t)$	$\mu^s_{6,1}(t)$	$\mu^s_{2,1}(t)$	$\mu^s_{4,1}(t)$	$\mu^s_{5,1}(t)$	α_1	α_2	α_3
0	0	0	0	0	0	0	0	0	0	0
1	0.0366	0	0.1291	0.4291	0.2100	0.0100	0.6600	0.0089	0.0286	0.0229
2	0.0970	0.0170	0.3305	0.6305	0.4543	0.2543	0.9043	0.0324	0.1019	0.0966
3	0.0835	0.0035	0.3083	0.6083	0.5688	0.3688	1	0.0527	0.1707	0.1941
4	0.1197	0.0397	0.4930	0.7930	0.7390	0.5390	1	0.0728	0.2530	0.3042
5	0.0779	0	0.5559	0.8559	0.7396	0.5396	1	0.0682	0.3070	0.3637
6	0.1237	0.0437	0.6873	0.9873	0.7394	0.5394	1	0.0780	0.3967	0.3993
7	0.0934	0.0134	0.5673	0.8673	0.7400	0.5400	1	0.0716	0.4197	0.3998
8	0.1031	0.0231	0.7245	1	0.7399	0.5399	1	0.0777	0.4628	0.3999
9	0.1393	0.0593	0.6940	0.9940	0.7401	0.7401	1	0.0815	0.4691	0.4002
10	0.1481	0.0681	0.7970	1	0.7408	0.5408	1	0.0947	0.5236	0.4005

Each entity has different goal so an entity sends recommendations to other entities based on its interest relationship with these entities. For example, entity α_2 has the same goal with entity α_5 so entity α_2 sends recommendations to entity α_5 with full accuracy. In contrast to entity α_5, entities α_2 and α_6 have different goals. Table 5.4 contains specific recommendations sent by entities α_1, α_2, and α_3 to other entities. Specifically, Table 5.4 shows initial values of specific recommendations sent to different entities by a specific entity. A specific recommendation on Table 5.4 contains the interest relationships between two entities.

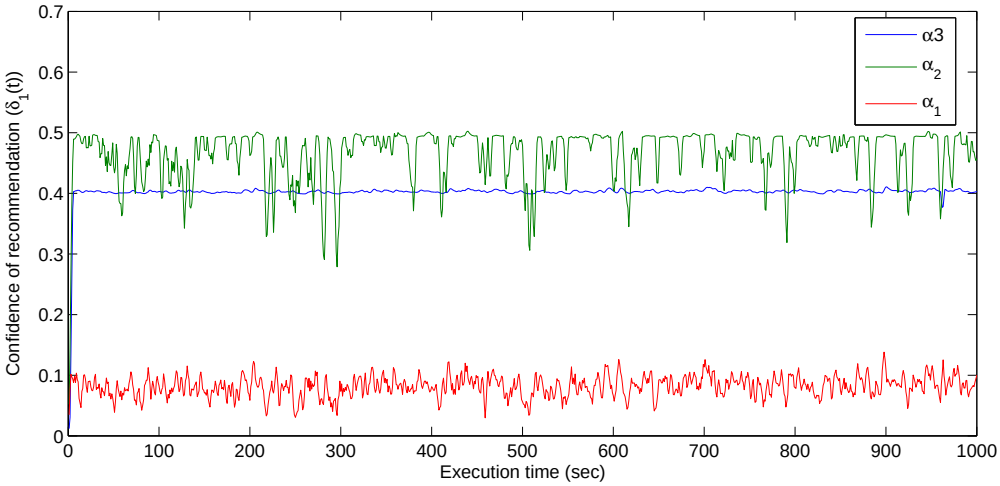

Figure 5.13. Confidences of recommendations $\delta_1(t)$ computed by entities α_1, α_2, and α_3 in long run.

The confidence of recommendation about the security system of service ω_i on an entity depends on ultimate experience $\epsilon_i(t)$ and ultimate recommendation $\mu_i(t)$ on the entity. Moreover, confidence of recommendation $\delta_i(t)$ depends also on reliability factor of service ω_i. A reliability factor about the security system of a service is specific to an entity and the reliability factor is constant. Therefore, confidence of recommendation $\delta_i(t)$ on an entity changes according to the ultimate experience $\epsilon_i(t)$ and ultimate recommendation $\mu_i(t)$. For example, Table 5.4 shows initial values of confidence of recommendation $\delta_1(t)$ computed by entities α_1, α_2, and α_3 in this case study. For the sake of simplicity, confidences of recommendations are computed by considering only the last three instances of ultimate experience $\epsilon_1(t)$ and ultimate recommendation $\mu_1(t)$ at a given time.

Table 5.4 shows that the behaviors of $\delta_1(t)$ computed by entities α_1, α_2, and α_3 are similar to the behaviors of $\mu_1(t)$ computed by entities α_1, α_2, and α_3. Additionally, the experimental results shown in Figure 5.13 verify the behaviors of $\delta_1(t)$ shown in Table 5.4. In this case study, a specific entity receives recommendations from other entities, where the entities obtain experiences from the same services with the specific entity. Therefore, the behaviors of confidences of recommendations are similar to the behavior of the ultimate recommendations as shown in Figure 5.13 and Figure 5.12. However, if a specific entity obtains experiences from some services that do not interact with entities, where the entities send recommendations to the specific entity, the behavior of confidences of recommendations may be different from the behaviors of ultimate recommendations.

The case study shows that flow of security evaluation information is a complex task that depends on many factors. Therefore, it is not realistic to represent security evaluation information with a single parameter. Our proposed model contains a number of factors related to the security system of a service in an entity and represents the properties of autonomous entities required for trust assessment in emerging networks. The experimental results clarify the behaviors of factors in the proposed model. Furthermore, the experimental results show that our model for flow of security evaluation information on entities reflects all properties of security evaluation information for assessing the trust of the security system of a service.

5.5.2.2. Trust Assessment. In this case study, the assessed trust is computed only on *Travel Agent* related to the security system of the online registration service of *Hotel A*. The assessments are carried out according to security evaluation information obtained over the flow network of security evaluation information on entities as shown in Figure 5.5.

We use the experimental results of the ultimate experience, the ultimate recommendation, and the confidence of recommendation about the security system of service ω_1 on entity α_2 for assessing trust. Moreover, we simulate algorithm 1 to assess trust

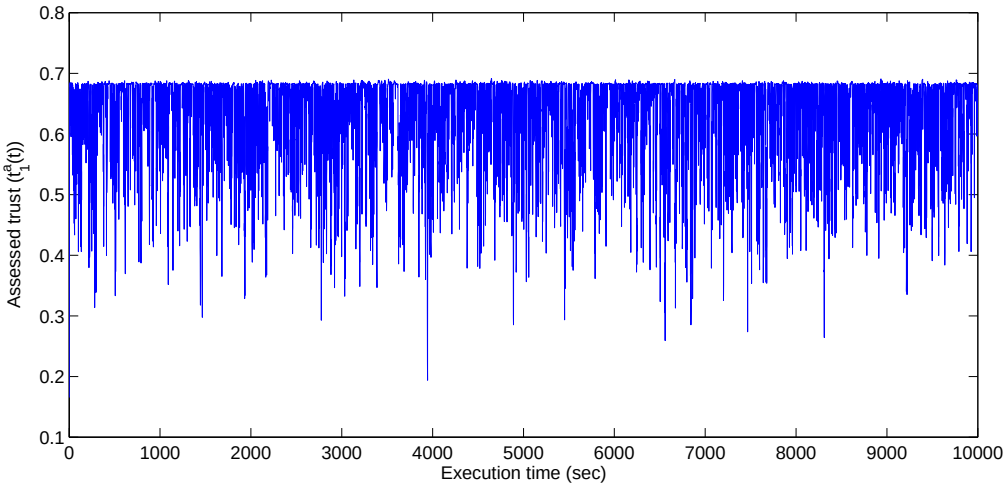

Figure 5.14. Assessed trust $tr_1^a(t)$ on entity α_2 in long run.

of the security system of service ω_1 on entity α_2. Figure 5.14 contains experimental results related to $t_1^a(t)$ on entity α_2 in long run. The value of the assessed trust depends on the security evaluation information because we compute the assessed trust by considering security evaluation information. For instance, Figure 5.15 contains the values of $t_1^a(t)$, $\epsilon_1(t)$, $\mu_1(t)$, and $\delta_1(t)$ on entity α_2 in long run.

The mean of obtained experiences is 0.5 in this case study. However, the mean of assessed trust $t_1^a(t)$ is 0.6321 that is computed by entity α_2. The mean of $t_1^a(t)$ is greater than the mean of obtained experiences because we consider older obtained experiences for computations of ultimate experience $\epsilon_1(t)$ and ultimate recommendation $\mu_1(t)$. Moreover, we take into account all received recommendations for the computation of ultimate recommendation $\mu_1(t)$. Due to the aging factor, both older experiences and older recommendations contribute to the computation of the assessed trust. Because entities obtain experiences from the same services, ultimate recommendation $\mu_1(t)$ behave as ultimate experience $\epsilon_1(t)$ in course of time. Therefore, the mean of assessed trust $t_1^a(t)$ is greater than the means of obtained experiences on entity α_2.

The maximum value of assessed trust $t_1^a(t)$ is 0.6915 computed by entity α_2, whereas the maximum value of obtained experiences is 1. Assessed trust $t_1^a(t)$ never has a value equal to 1 as shown in Figure 5.14 because of risk factors, accuracy factors of recommendations, and the distribution of the obtained experiences. Actually, the

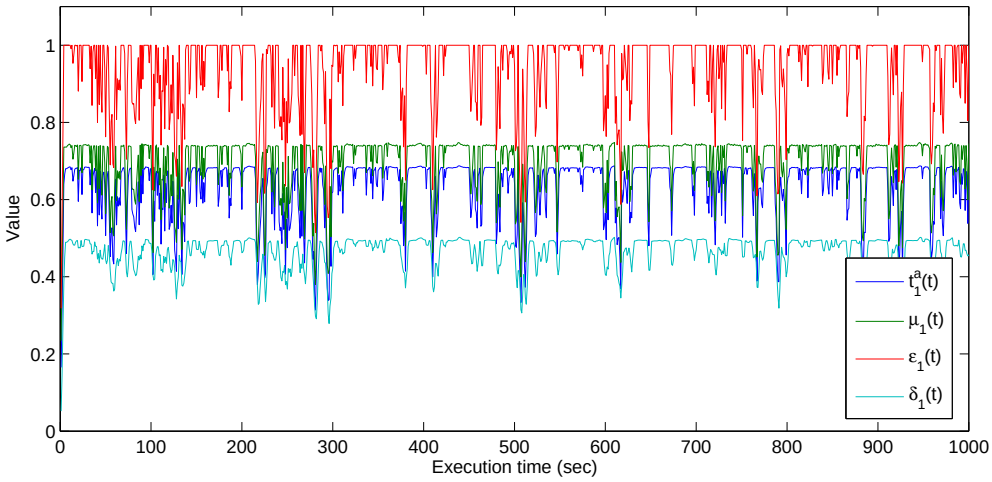

Figure 5.15. $\epsilon_1(t)$, $\mu_1(t)$, $\delta_1(t)$, and $t_1^a(t)$ on entity α_2 in long run.

risk factor and the accuracy factor reflect behaviors of autonomous entities when they obtain new data at a specific time. For instance, an entity does not trust to a service only with information obtained at a specific time.

A trust computation model has to consider the needs and the facts of a specific entity in emerging service oriented environments. Therefore, trust metrics related to the security system of a service has to be computed on each entity. On the other hand, security evaluation information must be propagated. However, each entity must compute its trust metrics individually and entities must not propagate the values of computed trust metrics to other entities. If trust is computed firstly in a source and then propagated to a destination, the destination will receive inaccurate trust values because of omitting real effects of intermediate nodes. Actually, it is not possible to know all facts and needs of autonomous entities. Therefore, computing trust metrics of an entity in a centralized manner is not realistic. Experimental results show that our model for flow of security evaluation information on entities provide high performance for obtaining security evaluation information about the security system of a service. Therefore, our trust assessment model better reflects the trust of an entity related to the security system of a service.

5.6. Conclusion

Emerging networks are expected to contain autonomous entities and support large number of various services. Moreover, the networks are expected to be open environments. Services in such environments should be accessible by entities and the entities should be able to interact with each other. Trust problems related to security systems of services create new research issues in emerging networks. In this chapter, we studied the issue of the trust assessment about the security system of a service according to needs of an entity. The trust assessment is based on flow of security evaluation information on entities.

Flow of security evaluation information is a significant task for the trust assessment in service-oriented environments. In this chapter, we proposed a crust model for flow of security evaluation information on entities. Additionally, we proposed a trust assessment model based on the security evaluation information flow model. The crust model uses direct security evaluation information and indirect security evaluation information of an entity about the security system of a specific service. In our crust model, experiences are direct security evaluation information, whereas recommendations and confidences of recommendations are indirect security evaluation information about the security system of a service.

In the flow model, an entity computes the ultimate experience about the security system of a specific service by considering its needs. The entity obtains experiences from a specific service. Additionally, the entity obtains experiences from other services that contain similar security systems with the security system of the specific service. All obtained experiences are aggregated by using similarity ratios. A similarity ratio represents the similarity between two security systems and is specific to an entity. The ultimate experience is computed according to aggregated experiences and the aging factor of older aggregated experiences. The aging factor is also specific to an entity and represents the subjective behavior of the entity to older experiences.

The ultimate recommendation about the security system of a service on an en-

tity is computed according to the ultimate experience, received recommendations from other entities, and confidences of recommendations about the security system of the service according to needs of the entity. Specifically, an entity receives recommendations about the security system of a service from different entities and aggregates them by using a risk factor. The risk factor shows the risk of using recommendations from an entity. The risk factor represents the belief of an entity about the recommendations received from a specific entity. The entity computes the ultimate recommendation according to the ultimate experience and total aggregated recommendations. The importance factor determines the contribution of the ultimate experience to the ultimate recommendation. Similar to the risk factor, the importance factor is specific to an entity. Moreover, an entity takes into account old ultimate recommendations to compute the recent ultimate recommendation according to the aging factor of recommendations. The aging factor of ultimate recommendations is also specific to an entity. Furthermore, an entity sends different recommendations to some other entities because of the utility relation between entities that is represented with an accuracy factor in our model.

The confidence of a recommendation about the security system of a service at an entity is computed according to the ultimate experience, the ultimate recommendation and the reliability factor. The reliability factor represents the subjective perception of the security system of a service for computations of confidences of recommendations on an entity.

In the trust assessment model, the assessed trust metric about the security system of a service is defined and computed as the mean of direct security evaluation information and indirect security evaluation information. Both direct security evaluation information and indirect security evaluation information are obtained according to flow of security evaluation information on entities.

The chapter contains the analysis of the crust model related to flow of security evaluation information on entities and the analysis of the trust assessment model. The online hotel reservation service was presented as a case study to demonstrate advantages of our flow model and trust assessment model. We simulated the case study

to analyze computations of the ultimate experiences, the ultimate recommendations, and confidence of recommendations on an entity. Moreover, we simulated the trust assessment model to clarify the pros of the trust assessment based on flow of security evaluation information. Performance analyses prove that our flow model consider subjective factors of an entity so the trust assessment model provides better trust assessment results about the security system of a service.

Our contribution is an entity model related to flow of security evaluation information on entities for trust assessment about the security system of a service for emerging networks and applications. Different from existing entity models, where trust is computed at each node, security evaluation information flows from sources to a destination on many entities without assessing trust in intermediate nodes. The trust assessment is accomplished only at the destination node. Moreover, our model for flow of security evaluation information contains subjective factors specific to an entity related to the security system of a service. The subjective factors reflect the behavior of entity on flow of security evaluation information and the trust assessment about the security system of a service. Our final contribution is the trust assessment model that is based on flow of security evaluation information on entities.

6. INTEROPERABILITY CRUST MODEL: TRUST BASED SECURITY INTEROPERABILITY FOR SERVICE CONVERGENCE IN NETWORKS

A security system is a set of security mechanisms. Security systems are interoperable if they contain similar security mechanisms. Similarity should be adequate for entities interacting with services. This chapter is about modeling security interoperability among convergent services in ubiquitous networks based on a new trust model. One problem of trust models is how to obtain sufficient information to compute the trust of a service. In existing trust computation approaches, an entity considers only its experiences about a service and recommendations of other entities. Such approaches are not adequate for critical services since trust computations take quite a long time. Determining trust values about critical services in a faster manner, such as assessing the trust on the security system of a service, is a significant problem. Also, the accuracy of trust computations is important. Having more experiences about the security system of a service affects trust computations. We present a crust model to increase security experiences related to a service to ensure fast and accurate trust computations. The results of trust computations are used for security interoperability decision making in networks. The chapter also contains a novel algorithm for security interoperability decisions based on trust. The case studies and simulations have shown that the proposed security experience gathering model significantly improves security experiences which results in fast and accurate security interoperability decision making.

The rest of the chapter is organized as follows. In Section 6.1, we present our motivation and contributions. We show related work in Section 6.2. Section 6.3 describes similarity based security experience aggregation from multiple services. We present trust computations and a decision algorithm for security interoperability of services in Section 6.4. We clarify contributions of the crust model via case studies and simulations in Section 6.5. Finally, we conclude this chapter in Section 6.6.

6.1. Introduction

Emerging networks are going to be heterogeneous networks. These networks are expected to interwork with each other and converge to a ubiquitous network, where the network is an open environment that supports large number of various services. Traditional network services are specific to networks. On the other hand, services in convergent networks must be accessible without considering the underlying access network. Service convergence necessitates managing services over a common environment with common mechanisms [114]. In addition, service convergence means that a user can use a service everywhere and everytime so that the quality of a service is expected to increase at a lower cost.

Security interoperability between services and entities is a significant issue to establish service convergence. Although the diversity of services increases in computer networks and service convergence becomes more important than ever, trust problems related to security issues become apparent. One such problem is the lack of information about the security of services. Specifically, an entity may have no information about security systems of a service in networks. Moreover, services may not provide detailed information related to their security systems to entities. Furthermore, security systems of services and entities may not fully comply with each other. On the other hand, an entity may still need to get a service from some services. Therefore, an entity should make trust assessments about security systems of services based on available security information about the services. If the security system of a service is not fully known by an entity, the entity can assess the trust of the security system based on available security information. So security interoperability between services and entities is ensured according to decisions based on the trust assessments.

In trust based systems, there are two main components, the source of trust information and the consumer of trust information. Trust information flows from a source to a destination according to a trust propagation model. Experience is trust information that flows from a source to a destination without passing over another party. If information flows from a source over another component or components to a destination,

such information is called recommendation.

6.1.1. Motivation

In computer networks, an entity obtains experiences and receives recommendations for computations of trust about services. Recommendations are gathered from many entities, whereas, experiences are obtained only from one source in traditional trust models. Since at least one additional entity has to participate to computations of recommendations, additional security problems occur in recommendation based trust models that is an undesirable situation for critical systems [70]. Therefore, experiences of an entity are preferred for computations of trust. For this reason, an entity needs to obtain as much as possible experiences in a short time. However, one source of experiences is usually inadequate to obtain accurate experiences for computations of trust that circumstance is a challenging problem for critical systems. So we need models to cope with such problem. Our motivation is the lack of a model to gather security experiences from services that have similar security systems for trust computations.

6.1.2. Contributions

In this chapter, we propose a model to obtain security experiences about the security system of a specific service from many services so we can compute the trust of a security system more accurately. In our model, security systems of services are similar to the security system of a specific service. Experiences are used for trust computations. The results of the trust computations are used for security interoperability decision making. We show the pros of the proposed model via case studies. Moreover, we analyze some parameters of the model with simulations. We can summarize the contribution of our work as follows.

- We introduce a formal model to obtain security experiences from a number of services for trust computations. The results of trust computations are used to make decisions in networks. For instance, trust computation results can be used for security interoperability decision making in convergent networks if there is a

lack of security evaluation information in such networks.

- We propose the concept of a similarity ratio for security systems of services. Moreover, we present a formal model for the computation of the similarity ratio. The similarity ratio enables aggregation of security experiences from many services related to the security system of a specific service. The similarity ratio approach improves the amount of security evaluation information related to the security system of a service. Therefore, security evaluation information based trust models will make more accurate trust computations.

- We introduce a new concept of awareness, namely service awareness. Service awareness is used to compute the quality of an ultimate experience. Based on the service awareness, we also introduce the quality of ultimate experience. The quality of ultimate experience provides more accurate trust computations.

- We present an algorithm for security interoperability decision making based on the trust assessment model. An entity can use the proposed algorithm to make security interoperability decision by considering its private needs.

6.2. Related Work

The amount of information affects the time needed to compute the level of trust. In existing trust computation models, an entity considers only its experiences about a service. Additionally, some models also use recommendations from other entities to compute trust. These models may be suitable for systems, where trust is determined after a long time. However, computing the trust of a critical system quickly is a significant problem.

Trust based systems usually combine general trust models according to their contexts and construct a trust network to propagate trust information from a source to a destination. Different types of trust propagation models are explored in [121]. Guha *et al.* propose a framework for propagation schemes of trust and distrust by considering

different circumstances [122]. A more specific study about the propagation of trust and distrust is accomplished by considering co-citations [120]. These researches do not contain models that aggregate direct information from different sources in each intermediate node. They compute trust metrics in each intermediate node and propagate values of trust metrics. Moreover, existing researches use models that consider only one source of direct information for trust computations.

Different from models mentioned above, our model obtains security experiences from similar security systems to compute the ultimate experience about a specific security system. Moreover, we consider only direct information between an entity and a service by omitting the trust propagation over a trust network because of security reasons mentioned in [70].

In literature, there are examples that use similarities for computations of recommendations [125–128]. Recommendations are computed by considering individual experiences of a node and by similar recommendations obtained from other nodes. Experiences obtained from multiple sources based on similarities are missing in these works. Our intention is to fill this gap.

6.3. Similarity Based Security Experience Aggregation

This section is an extension of Section 5.2.1 according to the similarity model proposed in this chapter. In this section, we present the model for aggregating security experiences from services that have similar security systems in networks. The aggregated security experiences are used for trust computations. Briefly, the experience aggregation is defined and explained more formally in this section to show how trust based security interoperability for service convergence in networks is ensured.

6.3.1. Environment

The environment is an overlay network containing services and entities over a convergent network. The environment consists of services and entities in a network.

Entities are autonomous agents or software applications representing users. Services are sources of security experiences. We assume that entities can obtain services only and they cannot provide any service. Detailed explanation and formal representation of the environment is given in Section 3.3.

6.3.2. Formal Representation of Security System

In this section, the security system of a service is formally represented with two sets of atomic units from the entity point of view. The first set is the set of atomic units of the security system and the second set is the expected set of atomic units of the security system.

6.3.2.1. Security System of a Service. In this chapter, the atomic unit representation of a security system is the same as the atomic unit representation of the security system of a service described in Section 3.4. We represent the security system of a service from the entity point of view with a set of atomic units, $\Phi(t) = \{\varphi_1, \ldots, \varphi_n\}$, where $n \in \mathbb{Z}^+$. An atomic unit of set $\Phi(t)$ is denoted with φ_i.

6.3.2.2. Expected Security System of a Service. Similar to Section 4.4.1, entities have expected security system representations of services. Specifically, an entity has a *set of expected atomic units* about the security system of a service. The entity needs to have experiences about all atomic units of the set. Set $\Phi_s^x(t) = \{\varphi_1, \ldots, \varphi_n\}$ represents the set of expected atomic units about the security system of service service ω_s and the set membership is time varying, where $n \in \mathbb{Z}^+$. For example, remember the security system representation of an airline reservation web service, $\Phi_{air}(t) = \{DES, AES128, AES256, DSS, PW\}$. Assume that an entity needs only authentication from the security system of the service so $\Phi_{air}^x(t) = \{DSS, PW\}$. In another case, assume that the entity needs 3DES and AES256 for encryption in addition to authentication mechanisms so $\Phi_{air}^x(t) = \{3DES, AES256, DSS, PW\}$. The set of expected atomic units and the set of atomic units of a service may be different.

The impact of an atomic unit is the effect of the atomic unit on computations of trust. If entities have the same representation of a security system, the entities have to consider the impact of each atomic unit. In this model, the impact of an atomic unit is related to the computations of experiences and similarity ratios.

The representation of expected security system of a service in Section 4.3.1 is different from the expected security system representation in this section. Security policies and security mechanisms are not distinguished in this section. Therefore, the set of expected security system of a service in this section is represented with $\Phi_s^x(t)$ related to service ω_s.

6.3.3. Obtaining Security Experiences

If an entity has previously interacted with services, the entity already has security experiences about the services. Experiences are information obtained by direct interactions of an entity with services. Therefore, an experience depends on a service and an entity.

Both an entity and a service can generate security information during an interaction as shown in Figure 6.1a. Actually, an entity or a service can have security information related to its security system and other security systems. Suppose that a service has a password based authentication system. It is expected that the service knows all properties of the authentication system, such as the acceptable length of a password. The password length information on the service is the security information generated on the service. On the other hand, suppose that an entity determines the acceptable length of a password for the service based on its interactions with the service. For example, the entity may have many successful and unsuccessful attempts to determine a password for authentication purpose with the service. The entity can determine the acceptable length of a password based on these attempts. The password length information related to the service is the security information generated on the entity. Briefly, both an entity and a service can generate security information.

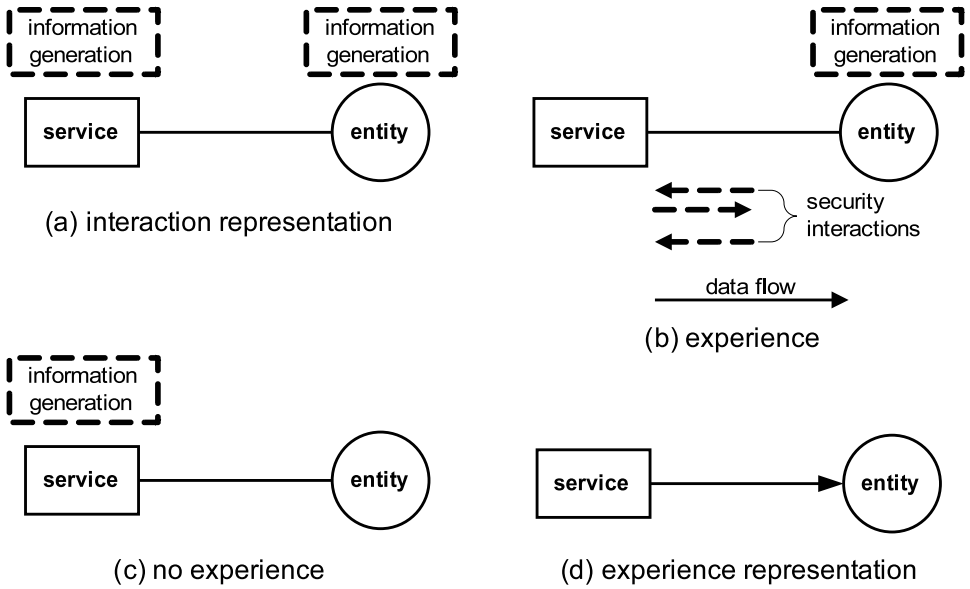

Figure 6.1. Experience obtained from a service by an entity.

Experiences are security information generated by entities themselves according to security data obtained after interactions with services as shown in Figure 6.1b. However, security information generated in a service is not an experience as shown in Figure 6.1c. In our model, the existence of an experience related to a service on an entity is represented as shown in Figure 6.1d. Figure 6.1 summarizes the way an experience is obtained. The formal definition of an experience is as follows:

Definition 6.1 *An experience is associated with security information generated by entity $\alpha \in A$ about the security system of service $\omega \in \Omega$ according to interactions between entity α and service ω.*

Set $V_s(t) = \{v_1(t), \ldots, v_n(t)\}$ represents the existence of experiences from the entity point of view about atomic units of set $\Phi_s(t)$, where $n \in \mathbb{Z}^+$, $\omega_s \in \Omega$, and

$$v_i(t) = \begin{cases} 1 & \textit{if experience about } \varphi_i \textit{ exists and} \\ & \textit{satisfies needs of the entity,} \\ 0 & \textit{otherwise.} \end{cases}$$

If $v_i(t) = 1$, then atomic unit φ_i is called *true experienced atomic unit*. In contrast, if $v_i(t) = 0$, then atomic unit φ_i is called *false experienced atomic unit*. For instance, consider the set of expected atomic units for airline ticket reservation service example, $\Phi^x_{air}(t) = \{3DES, AES256, DSS, PW\}$. Because the service does not contain 3DES, $v_{3DES}(t) = 0$. On the other hand, assume that the entity has information about AES256 that meets the needs of the entity so $v_{AES256}(t) = 1$. Assume also that the password based authentication mechanism does not satisfy the needs of the entity so $v_{PW}(t) = 0$. Additionally, assume that the entity has information about DSS that satisfies needs of the entity, therefore, $v_{DSS}(t) = 1$.

A false experienced atomic unit may be secure. For example, assume that a service has a password based authentication mechanism that is an atomic unit of the security system of the service. Assume also that an entity needs password based authentication mechanism for future transactions with the service. Additionally, the entity may request for the length of the password to be at least m characters. If the service allows passwords to be less than m characters and the entity senses this fact, the atomic unit is marked as a false experienced atomic unit. In this case, the authentication mechanism may be secure, but simply it does not satisfy the needs of the entity so that authentication mechanism is marked as a false experienced atomic unit.

$\Phi^r_s(t) = \{\varphi_x | v_x(t) = 1,\ \varphi_x \in \Phi_s(t) \wedge \varphi_x \in \Phi^x_s(t)\}$ represents *set of true experienced atomic units* about the security system of service ω_s. For instance, if we consider the security system of online ticket reservation service example, $\Phi^r_{air}(t) = \{AES256, DSS\}$.

$\Phi^f_s(t) = \{\varphi_x | v_x(t) = 0,\ \varphi_x \in \Phi_s(t) \wedge \varphi_x \in \Phi^x_s(t)\}$ is the *set of false experienced atomic units* about the security system of service ω_s, where $\Phi_s(t) \supseteq \Phi^r_s(t) \cup \Phi^f_s(t)$. For instance, if we consider the security system of online ticket reservation service example, $\Phi^f_{air}(t) = \{PW\}$.

An obtained experience is computed according to obtained security information about atomic units of the security system of a service. Formal definition of an obtained

experience is as follows:

Definition 6.2 *Obtained experience is a number* $\epsilon_s^r(t) \in [0,1]$ *representing the amount of experience about the security system of service* ω_s *in entity* α *that is obtained at the time* t*, where* $\omega_s \in \Omega$ *and* $\alpha \in A$*. An obtained experience is computed as follows, where* $|\Phi_s^x(t)| > 0,\ c^r > 0,\ c^f \geq 0,\ s \in \mathbb{Z}^+$.

$$\epsilon_s^r(t) = \max\left(0, \frac{c^r\left|\Phi_s^r(t)\right| - c^f\left|\Phi_s^f(t)\right|}{c^r\left|\Phi_s^x(t)\right|}\right). \tag{6.1}$$

We compute an obtained experience based on true experienced atomic units and false experienced atomic units of an entity about a service, normalized with respect to expected atomic units of the entity. Since entities are autonomous in our environment, they may be pessimistic or optimistic. Therefore, both true experienced atomic units and false experienced atomic units may have different impacts for the computation of an obtained experience. The coefficients c^r and c^f represent the autonomous nature of an entity for the computation of an obtained experience, where $c^r, c^f \in \mathbb{R}$. For example, if an entity is optimistic, the entity is expected to have $c^r > c^f$. In our model, we assume $c^r = 1$ and $c^f = 1$.

If an entity has experiences about all atomic units of the security system of service ω_s and the atomic units satify needs of the entity, then obtained experience $\epsilon_s^r(t) = 1$. On the other hand, if there is no information about any atomic unit of the security system or existing information about atomic units do not satisfies needs of the entity, then obtained experience $\epsilon_s^r(t) = 0$. Moreover, $0 < \epsilon_s^r(t) < 1$ means that the security system partially satisies needs of the entity. For instance, obtained experience $\epsilon_{air}^r(t) = 0.25$ for the security system of the ticked reservation service example, where $|\Phi_{air}^r(t)| = 2$, $\left|\Phi_{air}^f(t)\right| = 1$, and $|\Phi_{air}^x(t)| = 4$.

6.3.4. Similarity of Security Systems

Experiences of an entity about a service may affect the amount of experiences about another service. On the other hand, the security system of a service may be similar or the same with the security system of another service. Let sets $\Phi_a(t)$ and $\Phi_b(t)$ represent security systems of services ω_a and ω_b respectively. Sets $\Phi_a(t)$ and $\Phi_b(t)$ may contain the same atomic units but are not expected to be the same. An entity can use experiences about service ω_a to figure out the amount of experiences about service ω_b if set $\Phi_a(t)$ is similar to set $\Phi_b(t)$. In our model, experiences about a service are used to determine the amount of experiences about another service according to a similarity ratio.

A similarity ratio represents the similarity between security systems of two services. Similarity ratio λ_a^b means that the security system of service ω_a is similar to the security system of service ω_b. The value of λ_a^b closer to one means the security system of service ω_a is more similar to the security system of service ω_b. In addition, zero means no similarity and one means completely identical security systems.

A similarity ratio is asymmetric. For instance, similarity ratio λ_a^b is not the same with similarity ratio λ_b^a. Assume that service ω_a has only one security property, which is represented with a single atomic unit in an entity. On the other hand, assume that the security system of service ω_b is represented with five atomic units in the entity. Assume also that the security system of service ω_b contains the security property of service ω_a. In this case, it is expected that $\lambda_b^a > \lambda_a^b$ because the security system representation of service ω_b has more atomic units than the security system representation of service ω_a and the atomic unit of service ω_a is also an atomic of the set representation of service ω_b.

There are three factors that affect the value of a similarity ratio. Each similarity ratio is computed according to needs of a specific entity so a similarity ratio is specific to the entity. Therefore, the first factor is an entity. The second factor is the security system of a service which is compared with the security system of a specific service.

The last factor is the security system of a service to which other security systems are referred. For instance, similarity ratio λ_a^b in an entity is computed according to security systems of services ω_a and ω_b, and needs of the entity. The formal definition of the similarity ratio is as follows:

Definition 6.3 *Similarity ratio is a number $\lambda_x^y(t) \in [0,1]$ representing the similarity of the security system of service ω_x to the security system of service ω_y according to needs of entity α at a given time t, where $\omega_x, \omega_y \in \Omega$, $\alpha \in A$ and is computed with the similarity function of the entity. The similarity function of an entity represents the needs of the entity. The notation of the similarity function of entity α is as follows.*

$$\lambda_x^y(t) = f_\alpha^{sim}\left(\Phi_y(t), \Phi_x(t)\right)(t). \tag{6.2}$$

A similarity ratio is determined according to atomic units of security systems of two services. Each entity may have different similarity functions for computing the values of similarity ratios, but all entities have to compute their similarity ratios according to atomic units of two security systems. An example similarity function of entity α is shown below.

$$f_\alpha^{sim}\left(\Phi_y(t), \Phi_x(t)\right)(t) = \frac{|\Phi_y(t) \cap \Phi_x(t)|}{|\Phi_y(t)|}. \tag{6.3}$$

Assume that an entity has a similarity function as Equation 6.3 and needs to compute similarity ratios $\lambda_x^y(t)$ and $\lambda_y^x(t)$ as shown in Figure 6.2, where $|\Phi_y(t)| > 0$. Let $\{DES, 3DES, DSS\}$ and $\{DES, AES128, AES256, DSS, SHA1\}$ be sets $\Phi_x(t)$ and $\Phi_y(t)$ in sequence. Then, similarity ratios $\lambda_x^y(t) = \frac{2}{5}$ and $\lambda_y^x(t) = \frac{2}{3}$ in the entity.

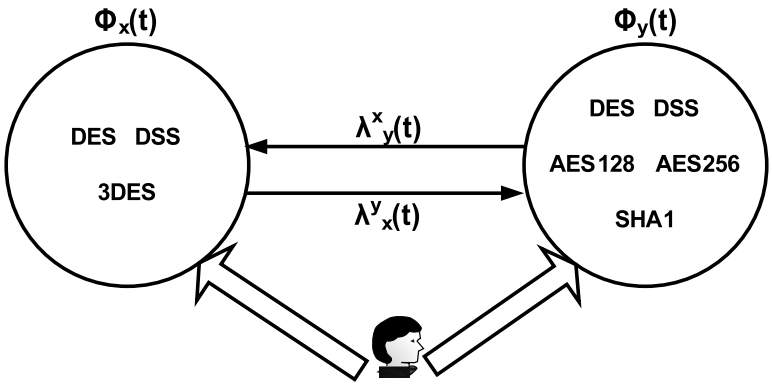

Figure 6.2. Similarities of two security systems in an entity.

Note that $\lambda_m^i(t) > \lambda_n^i(t)$ means the security system of service ω_m is more similar than the security system of service ω_n to the security system of service ω_i.

6.3.5. Aggregating Experiences: Perceived, Aggregated, and Ultimate Experiences

An entity may have obtained experiences from many services. Obtained experiences from services contribute differently to the computation of the ultimate experience. The contribution of an obtained experience depends on the perception of an entity. Several entities may have the same obtained experience related to the security system of a specific service. However, all entities perceive the experience differently because of the difference among their similarity functions. *Perceived experience* represents the experience perceived by an entity about the security system of a specific service. The formal definition of the perceived experience is as follows.

Definition 6.4 *Perceived experience about the security system of service ω_j related to the security system of service ω_s is a number $\epsilon_j^p(t) \in [0,1]$ representing the perception of an obtained experience $\epsilon_j^r(t)$ in entity α, where $\omega_j, \omega_s \in \Omega$ and $\alpha \in A$. Perceived experience $\epsilon_j^p(t)$ is the obtained experience $\epsilon_j^r(t)$, weighted by the similarity ratio $\lambda_j^s(t)$ as follows.*

$$\epsilon_j^p(t) = \epsilon_j^r(t)\,\lambda_j^s(t), \quad s, j \in \mathbb{Z}^+. \tag{6.4}$$

Consider obtained experience $\epsilon_{air}^r(t) = 0.25$ for computation of perceived experience $\epsilon_{air}^p(t)$ related to the online ticket registration service example. Suppose that the similarity ratio for the security system of the ticked reservation service is $\lambda_{air}^s(t) = 0.4$. Then, perceived experience $\epsilon_{air}^p(t) = 0.1$. The value of a perceived experience never exceeds the value its related obtained experience because the value of a similarity ratio can be at most one.

Entity α may interact with different services in course of time. Set $\Omega_\alpha(t)$ represents the services interacting with entity α in the entity and the set is dynamic. The entity aggregates perceived experiences at a given time according to elements of set $\Omega_\alpha(t)$. Entities may not know the whole environment. Therefore, entities construct a directed acyclic graph of services to obtain a specific security experience at a particular time.

Since an entity can interact with many services, it is not realistic to take into account all services it interacts for the computation of an aggregated experience. One of the reasons is that a network may contain devices with limited resources, such as a limited power or a low computation capability. However, an entity running on these devices still has to be able to compute aggregated experiences. Therefore, an entity determines a *maximum number of services* from which the entity can receive security data. The maximum number may change over time. We represent the maximum number of services in an entity with $W(t) \in \mathbb{Z}^+$. Number $W(t)$ also shows the *required number of services* to compute a reliable aggregated experience. Therefore, the aggregated experience is normalized with $W(t)$.

Services affect the value of the aggregated experience. Therefore, entity α must choose services it interacts to maximize its aggregated experience. The *maximum*

interacted set consists of services that maximize the aggregated experience of entity α and the set is represented with $\Omega_{\alpha}^{max}(t) \subseteq \Omega_{\alpha}(t)$. Elements of set $\Omega_{\alpha}^{max}(t)$ are determined according to a utility function, where the output of the function represents the utility relation between the entity and a service. Each entity may have different utility function to determine maximum interacted set. The set is computed as follows.

$$\Omega_{\alpha}^{max}(t) = \left\{ \begin{array}{l} \omega_y | \omega_y \in \Omega_{\alpha}(t), \\ \omega_y = \arg\max_{\omega_s} \; u_{\alpha}(\omega_s), \\ s, y \in \mathbb{Z}^{+}. \end{array} \right\} \tag{6.5}$$

$u_{\alpha}(\omega_s)$ is the *utility function of entity* $\alpha \in A$ that takes into account service ω_s for selecting services. If an entity has obtained experiences from more than $W(t)$ services, then the entity selects services that maximize its utility function as elements of set $\Omega_{\alpha}^{max}(t)$. Based on the maximum interacted set, the formal definition of aggregated experience is as follows.

Definition 6.5 *Aggregated experience about the security system of service* ω_s *in entity* α *is weighted sum of perceived experiences at time* t*, where* $\omega_s \in \Omega_{\alpha}(t)$ *and* $\alpha \in A$*. The aggregated experience is represented with* $\epsilon_s^a(t) \in [0,1]$ *and is computed as follows, where* $W(t) > 0$ *and* $W(t) \geq |\Omega_{\alpha}^{max}(t)|$*.*

$$\epsilon_s^a(t) = \frac{1}{W(t)} \sum_{\forall \omega_j \in \Omega_{\alpha}^{max}(t)} \epsilon_j^p(t). \tag{6.6}$$

Suppose that an entity has perceived experiences from services ω_1 and ω_2 related to the security system of service ω_s, where $\epsilon_1^p(t) = 0.7$ and $\epsilon_2^p(t) = 0.5$. Suppose also that the required number of services for the computation of aggregated experience $\epsilon_s^a(t)$

is 3 in the entity, $W(t) = 3$. In this case, aggregated experience $\epsilon_s^a(t) = 0.4$.

Entities are continuously interacting with services. Therefore, an entity may have different aggregated experiences for a specific time instance, where the aggregated experience is independent from other aggregated experinces. However, all aggregated experiences have to be represented during computations of trust metrics in a trust computation model. For this reason, we introduce the *ultimate experience* about the security system of a service, where the effect of time is considered. The formal definition of ultimate experience is as follows.

Definition 6.6 *Let entity α has been interacting with service ω_s for x time slots, where $\alpha \in A$ and $\omega_s \in \Omega$. Let $\epsilon_s^a(t)$ represents an aggregated experience about service ω_s at the time t. Then, the ultimate experience about the security system of service ω_s in entity α is the weighted sum of all aggregated experiences until the time t, where aging factor τ determines the weight of an aggregate experience. The ultimate experience is represented with $\epsilon_s(t) \in [0, 1]$ and is computed as below.*

$$\epsilon_s(t) = min\left(1, \sum_{x=0}^{t-1} \tau^{x+1} \epsilon_s^a(t-x)\right). \tag{6.7}$$

Assume that an entity has aging factor $\tau = 0.5$ for computing ultimate experiences. Assume also that the entity has four aggregated experiences related to service ω_s, which aggregated experiences are $\epsilon_s^a(t) = 0.4$, $\epsilon_s^a(t-1) = 0.8$, $\epsilon_s^a(t-2) = 0.6$, and $\epsilon_s^a(t-3) = 0.3$. Based on these facts, the entity can compute ultimate experience $\epsilon_s(t)$, where ultimate experience $\epsilon_s(t) = 0.49375$.

Ultimate experiences computed according to Equation 6.7 give more weights to newly computed aggregated experiences than older experiences. The time at which experiences are aggregated may be significant for computations of ultimate experiences.

For example, the last aggregated experiences may be more significant than older ones. In this case, an entity may compute ultimate experiences by using a constant aging factor specific to the entity as in Equation 6.7. The aging factor is represented with $\tau \in [0, 1]$.

Entities may evaluate aggregated experiences in a different manner by considering its history of experiences, where aging factor τ is time dependent, $\tau(t) \in [0, 1]$. For instance, experiences aggregated at specific times may be more significant than others, such as if the entity had security experiences from two significant services acquired at a specific time, these experiences may have more weight for computing the ultimate experience in similar situations. In this case, the ultimate experience about service ω_s is computed with Equation 6.8.

$$\epsilon_s(t) = min\left(1, \sum_{x=0}^{t-1} \tau(t)\, \epsilon_s^a(t-x)\right). \tag{6.8}$$

The value of an aging factor determines the contribution of an aggregated experience to the ultimate experience. For example, aging factor $\tau(t_a) = 0.5$ contributes more to the computation of ultimate experiences than aging factor $\tau(t_b) = 0.3$.

6.4. Trust Computation and Security Interoperability

In this section, we explain our trust computation model. Moreover, we propose an algorithm for security interoperability decision making.

We define *trust* as the security expectation of an entity from a service according to available security information about that service on the entity. The available security information are security experiences of an entity about the security system of a service and the quality of the experiences. The quality of an experience depends on the entity's awareness about each service from which security experiences are obtained.

6.4.1. Service Awareness

An entity may or may not identify a service. For example, a service may have a certificate from a known certificate authority and an entity may identify the service by using the certificate. On the other hand, the entity may not have sufficient information about the service so it may be impossible to identify the service. An entity may know a service, but the entity may have no trust to the service for many reasons, such as a contradiction between the goal of the entity and the goal of the service.

The identification of a service by an entity is significant for computations of trust metrics related to that service. Therefore, an entity must be aware of services from which the entity obtains experiences. In our model, service awareness is defined as follows.

Definition 6.7 *Service awareness is a number* $k_s^{\omega}(t) \in \{0, 1\}$ *that represents the identity information about service* ω_s *in entity* α *at a given time* t*, where* $\omega_s \in \Omega$ *and* $\alpha \in A$*.*

If entity α believes that it has sufficient information about the identity of service ω_s at a given time t, service awareness $k_s^{\omega}(t) = 1$, otherwise service awareness $k_s^{\omega}(t) = 0$. For example, assume that entity α needs to identify service ω_s, but initially the entity has no identity information related to the service. In this case, service awareness $k_s^{\omega}(t) = 0$ in the entity. The entity requests identity information from the service. The service submits its identity information the entity based on the request. The identity information contains a valid certificate from a certification authority recognized by the entity. Therefore, the entity changes the value of service awareness $k_s^{\omega}(t)$ to one.

6.4.2. Service Reputation

A network can contain lots of services. Therefore, the reputation of services that contribute to compute trust metrics related to a specific service is significant and is defined as follows.

Definition 6.8 *Service reputation is a number* $r_s(t) \in [0,1]$ *that represents the general belief of entity* α *about service* ω_s *at a given time* t*, where* $\alpha \in A$ *and* $\omega_s \in \Omega$*.*

Larger values of service reputation $r_s(t)$ mean that services have better reputations. For instance, if service reputation $r_s(t) = 1$, services have the best reputation, whereas if service reputation $r_s(t) = 0$, services have the worst reputation. The computation of service reputation is out of scope of this thesis and values of service reputations are arbitrarily assigned.

6.4.3. Quality of Ultimate Experience

Based on our definitions of service awareness and service reputation, we are now ready to define the quality of an ultimate experience, which is to be used to compute a trust metric. We consider services that contribute to the computation of the ultimate experience related to the computation of the quality of an ultimate experience.

Definition 6.9 *The quality of the ultimate experience about service* ω_s *in entity* α *is a number* $k_s(t) \in [0,1]$ *that represents the impact of ultimate experience* $\epsilon_s(t)$ *and is compued as follows, where* $\omega_s \in \Omega$*,* $\alpha \in A$*,* $|\Omega_\alpha^{max}(t)| > 0$*, and* $x, s \in \mathbb{Z}^+$*.*

$$k_s(t) = min\left(1, \left(\frac{1}{|\Omega_\alpha^{max}(t)|}\sum_{\forall \omega_x \in \Omega_\alpha^{max}(t)} k_x^\omega(t)\right) + r_s(t)\right). \tag{6.9}$$

Assume that an entity computes ultimate experience about service ω_s by using experience from five services, where $|\Omega_\alpha^{max}(t)| = 5$. The entity identifies three of these services. Moreover, reputation of the service $r_s(t) = 0.2$. Therefore, the quality of ultimate experience $k_s(t) = 0.5$.

6.4.4. Computation of Trust

An entity has two trust metrics for security interoperability decision making. The first metric, namely *assessed trust* represents the trust assessed by an entity about the security system of a service. The second trust metric is *minimum trust* about the security system of a service on an entity. The security system of the service has to have a minimum trust value for an acceptable security interoperability decision. The minimum trust is represented with $tr_s^m(t)$, where $tr_s^m(t) \in [0,1]$ and $\omega_s \in \Omega$. Due to the autonomous nature of entities, each entity may have different minimum trust values. Each entity can assess trust differently according to its utility function, the ultimate experience, and the quality of the ultimate experience. Therefore, the formal definition of the assessed trust is as follows.

Definition 6.10 *Let $u_{trust}(\epsilon_s(t), k_s(t))(t)$ be the utility function of entity α for trust assessment, where $\alpha \in A$ and $\omega_s \in \Omega$. Let $\epsilon_s(t)$ be the ultimate experience and $k_s(t)$ be the quality of the ultimate experience. Then, the assessed trust on entity α about the security system of service ω_s is a number $tr_s^a(t) \in [0,1]$ and is computed with utility function u_{trust} of entity α.*

An entity can compute assessed trust values for each service it interacts so the entity can have different assessed trust values at the same time. However, an entity has only one utility function for computations of assessed trust metric related to all services. An example utility function of an entity may be as follows.

$$u_{trust}(\epsilon_s(t), k_s(t))(t) = \epsilon_s(t)\, k_s(t). \tag{6.10}$$

Actually, each entity may have different trust representations, such as one may represent trust with a totally ordered set of nonnegative integers within a certain

range, another may represent trust with continuous numbers. In our model, however, we assume that values of trust metrics are normalized and their ranges are the same as the range of the minimum trust.

6.4.5. Decision of Security Interoperability

Our central goal is to be able to make a security interoperability decision based on trust metrics that we have introduced. Algorithm shown in Figure 6.4.5 shows how a security interoperability decision is made about the security system of a service based on trust assessment. In brief, the algorithm performs the following steps. An entity computes the assessed trust about the security system of a service. Then, the entity compares the assessed trust with the minimum trust. If the assessed trust is less than the minimum trust, the security interoperability decision is called "Not Satisfactory", meaning the security system of the service does not satisfy the needs of the entity. Otherwise, the security interoperability decision is called "Satisfactory", meaning the security system of the service satisfies the needs of the entity.

6.5. Case Studies and Simulations

We illustrate the advantage of our proposed model with three case studies and simulations. In first case study, we investigate the impact of multiple sources of security experiences. In second case study, steps of a trust assessment and the performance evaluation of aggregating similar security experiences are presented. The last case study is about a security interoperability decision making for service convergence in networks.

In these case studies, time is discrete and increases every second by one. Specifically, an entity obtains security experiences every second and completes computations within one second. Simulations were carried out by using MATLAB R2009b version 7.9.0.529 that runs on a PC with Intel Core 2 Duo E8400 3.00GHz processor and 3GB of RAM.

Require: $|\Omega_{\alpha}^{max}(t)| > 0$, $W(t) > 0$, $|\Omega_{\alpha}^{max}(t)| \leq W(t)$

Ensure: Security interoperability decision for ω_s;

1: $all\epsilon_j^p(t) \leftarrow 0$; represents the sum of all perceived experiences
2: **for all** ω_j such that $\omega_j \in \Omega_{\alpha}^{max}(t)$ **do**
3: $\epsilon_j^p(t) \leftarrow \epsilon_j^r(t) \lambda_j^s(t)$;
4: $all\epsilon_j^p(t) \leftarrow all\epsilon_j^p(t) + \epsilon_j^p(t)$;
5: **end for**
6: $\epsilon_s^a(t) \leftarrow all\epsilon_j^p(t) / W(t)$;
7: $\epsilon_s(t) \leftarrow 0$;
8: **for** $x = 0$ to t **do**
9: $\epsilon_s(t) \leftarrow \epsilon_s^a(x) \tau(t)$;
10: **end for**
11: **if** $\epsilon_s(t) > 1$ **then**
12: $\epsilon_s(t) \leftarrow 1$;
13: **end if**
14: $tr_s^a(t) \leftarrow u_{trust}(\epsilon_s(t), k_s(t))(t)$;
15: **if** $tr_s^a(t) < tr_s^m(t)$ **then**
16: the security system of service ω_s does not satisfy the needs of the entity (Not Satisfactory);
17: **else**
18: the security system of service ω_s satisfies the needs of the entity (Satisfactory);
19: **end if**

Figure 6.3. The Algorithm for Trust Based Security Interoperability.

6.5.1. Case Study 1: Gathering Security Experiences from Multiple Services

In this case study, an entity that represents an engineering student at Bogazici University (BU) requests help from an instructor about the preparation of a term project. Instructors at BU can help students by an online service only. Additionally, a student has to complete some initial steps of the term project and submit it to the online service of an instructor to be able to request online help from the instructor.

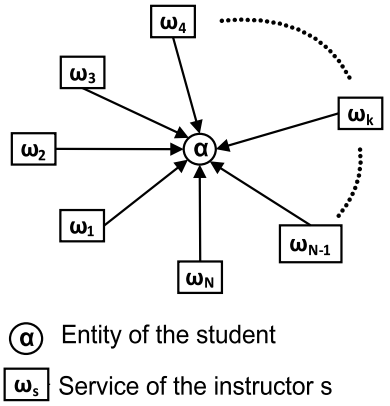

Figure 6.4. A network for security experience aggregation from multiple services.

On the other hand, there are naughty students who look for hard-working students who may submit their projects to the naughty students' fake online services. The aim of a naughty student is to steal hard-working students' partially completed projects and to get help from an instructor. If a hard-working student submits a partially completed term project to a fake online service, the student will get a low grade from the term project and the naughty student will get a high grade.

Each online service of instructors has similar security systems. But security systems of fake online services differ from security systems of instructors' online services. For this reason, a hard-working student should assess the trust of the security system of an instructor's online service. Therefore, the engineering student gathers security experiences about the security system of the instructor's online service. The student also considers security systems of some other instructors' online services. The student can access instructors' online services from many networks. Instructors' online services

can run on any computing platform, such as on a laptop computer. Figure 6.4 shows the logical network of this case study.

The aim of this case study is to show effects of similar services for the computation of the ultimate experience according to different aging factors. Since the ultimate experience is directly proportional with the aggregated experience, here effects of many services on the aggregated experience are also shown.

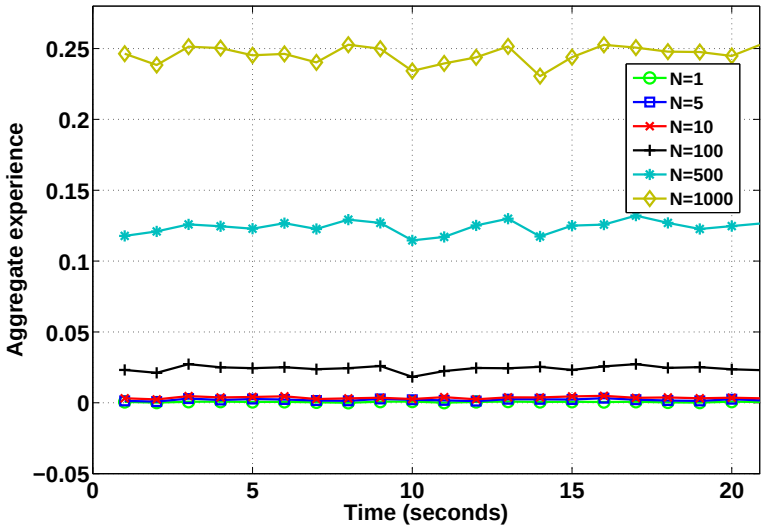

Figure 6.5. Initial few steps of the computation of the aggregated experience in the entity with different N.

6.5.1.1. Assumptions. For the sake of brevity, we set some parameters of the proposed model to be constant. Maximum number of services $W(t) = 1000$ and it does not change with time. The elements of set $\Omega_{\alpha}^{max}(t)$ are constant during a simulation. We represent the number of elements of set $\Omega_{\alpha}^{max}(t)$ with $N = |\Omega_{\alpha}^{max}(t)|$. The entity can compute all similarity ratios for all services. Moreover, similarity ratios do not change with time and they are uniformly distributed with mean equal to 0.5. All obtained experiences are uniformly distributed with mean equal to 0.5.

6.5.1.2. Performance Evaluation. Although we assume that both similarity ratios and obtained experiences are uniformly distributed with mean equal to 0.5, the expected value of the aggregated experience has to be 0.25, if W=N. The simulation results shown in Figure 6.5 and Figure 6.6 verify this assertion.

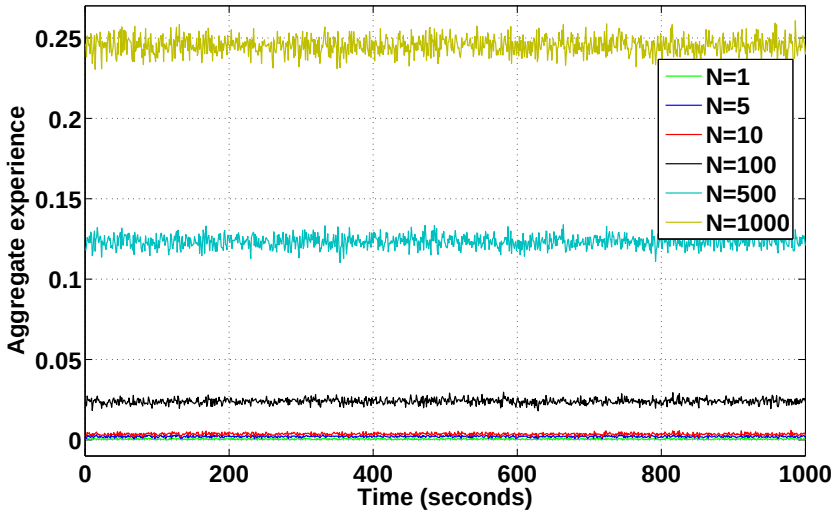

Figure 6.6. The computation of the aggregated experience in the entity with different N in long run.

For smaller values of N, such as $N = 1$ and $N = 5$, the aggregated experience approaches to zero. On the other hand, for larger values of N that are close to W, such as $N = 1000$, the aggregated experience approaches to 0.25. The simulation results in long run also prove that the aggregated experience converges to zero for smaller values of N and the aggregated experience converges to 0.25 for larger values of N as shown in Figure 6.6.

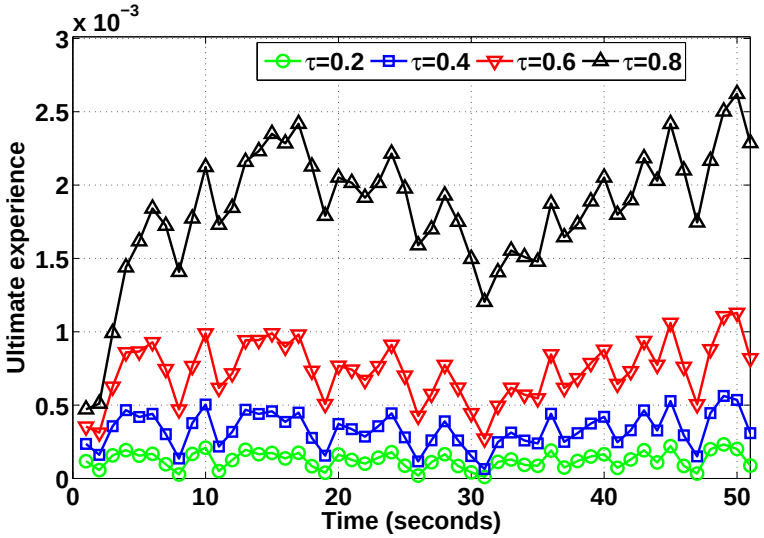

Figure 6.7. The initial behavior of the ultimate experience under different aging factors in case of one service.

Because the entity needs a certain amount of security experiences to reach a particular aggregate experience level, it takes time to obtain the amount of security experiences with smaller values of N. Additionally, when an entity obtains the desired amount of security experiences, some experiences are aging and they do not provide precise information for the computation of the ultimate experience. Therefore, the ultimate experience never reaches to higher levels with smaller values of N as shown in Figure 6.7 and Figure 6.8.

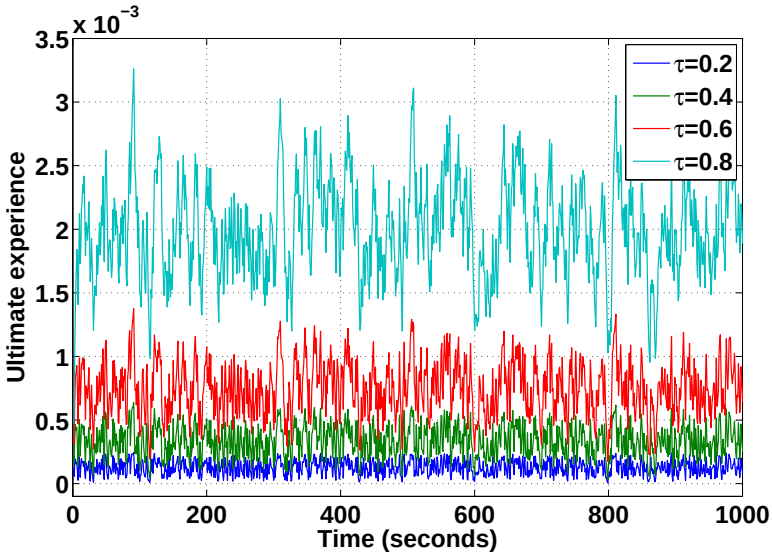

Figure 6.8. The behavior of the ultimate experience under different aging factors in case of one service in long run.

For the sake of simplicity, we chose Equation 6.7 for the computation of an ultimate experience. According to the Equation, the behavior of an ultimate experience depends on aggregated experiences and aging factors. If $N << W$, ultimate experiences behave as shown in Figure 6.7 and Figure 6.8. On the other hand, if $N \cong W$, ultimate experiences are as shown in Figure 6.9 and Figure 6.10.

Figure 6.7 and Figure 6.8 show the behavior of the ultimate experience under four different values of aging factors if there is only one service. All computations of the ultimate experience converge to a value approximately after five steps as shown in Figure 6.7. Additionally, the ultimate experience converges faster for smaller values of the aging factor than larger values of the aging factor as shown in Figure 6.8. In each new step, the effect of an aging factor on a specific aggregated experience at certain

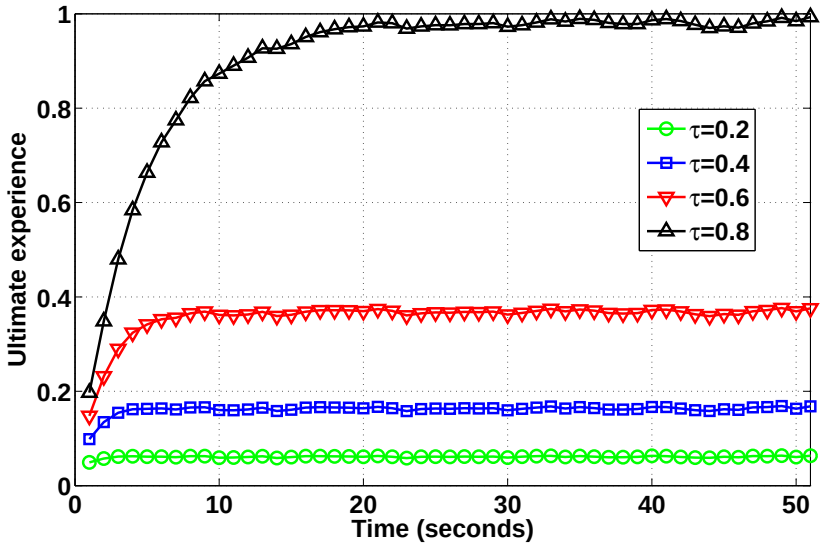

Figure 6.9. The initial behavior of the ultimate experience under different aging factors based on many services.

time increases so an obtained experience does not have remarkable influence on the ultimate experience after many times. On the other hand, the variance of the ultimate experience is relatively high if $N << W$ because the ultimate experience depends on a limited number of services.

If $N \cong W$, then the variance of the ultimate experience is relatively small as shown in Figure 6.9 and Figure 6.10. However, the ultimate experience converges slower for larger values of the aging factor than smaller values because obtained experiences do not get older as fast as in $N << W$ case. For instance, the ultimate experience converges approximately to one after 20 steps for the aging factor equal to 0.8, but it converges to 0.15 after three steps for the aging factor equal to 0.2 as shown in Figure 6.9.

Simulation results show that an aging factor is significant for the computation of an ultimate experience so the aging factor affects the assessed trust in an entity. Additionally, larger values of aging factors are better to get larger values of ultimate experiences. Therefore, an aging factor with larger value ensures more accurate trust assessment results. Similar to an aging factor, the simulation results show that multiple services are better than a single service to compute an ultimate experience.

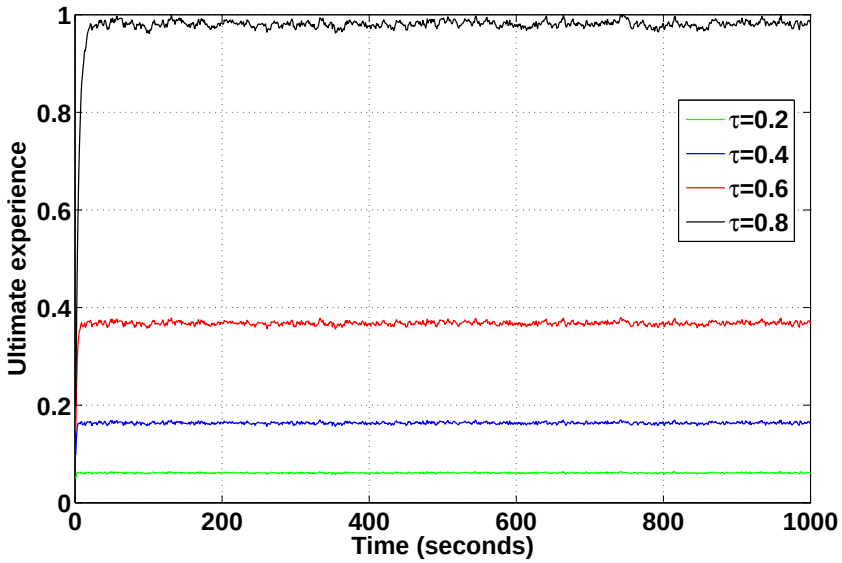

Figure 6.10. The behavior of the ultimate experience under different aging factors based on many services in long run.

6.5.2. Case Study 2: Trust Computation for a User Accessing Multiple Services

A professor at Bogazici University will attend a conference in a foreign country and she wants to register to hotel C that is close to the conference location by using the online registration option of the hotel. Moreover, the professor knows that there are some hotels which have weak security systems in their online registration services. If the professor uses an online registration service with weak security system, she may lose some confidential information so she may have financial problems. Therefore the professor needs to trust to the security system of the online registration service before registering to hotel C.

The professor has a mobile phone on which there is an intelligent agent searching for the best accommodation place. Additionally, the agent is capable to carry out the trust assessment about the security system of an online registration service. Figure 6.11a shows the information gathering scenario for trust assessment. The professor is represented with User in Figure 6.11a who wants to register to hotel C. Logical representation of this scenario is shown in Figure 6.11b, where entity α represents the

agent on the mobile phone of the professor. The online registration service of hotel C is represented with service ω_3.

The entity needs to carry out a trust assessment about the security system of service ω_3 for online registration. However, the entity does not have enough security information to carry out such trust assessment. For this reason, entity α collects security experiences from many other online services that contain similar security systems to the security system of service ω_3. Because of limited resources of the mobile phone, the entity selects only limited number of online services that have the most similar security systems to the security system of service ω_3.

In this case study, the trust assessment is accomplished in two different scenarios. In the first scenario, only security experiences from service ω_3 is considered whereas in the second scenario, entity α considers security experiences from similar services in addition to security experiences from service ω_3.

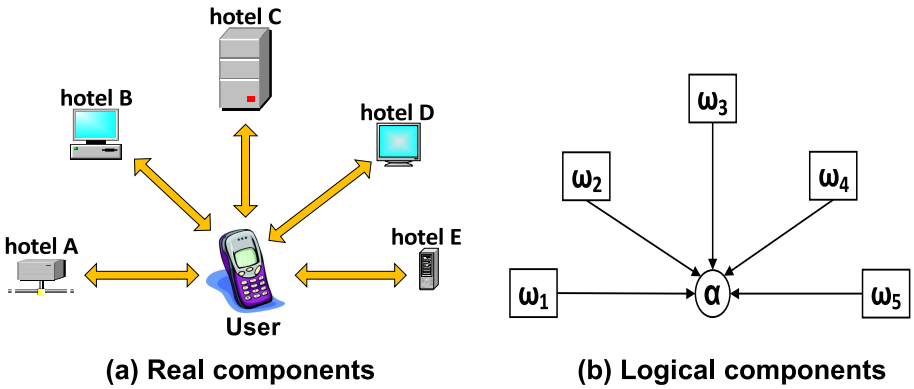

Figure 6.11. Information gathering scenario for trust computations.

Briefly, entity α wants to get a service from service ω_3, but before getting the service, it needs to carry out a trust assessment about the security system of the service for future interactions.

6.5.2.1. The Facts for Trust Computation. The maximum required number of services is 5, $W(t) = 5$, and does not change with time. The elements of set $\Omega_\alpha^{max}(t)$ are constant during a simulation. We represent the number of elements of set $\Omega_\alpha^{max}(t)$ with

$N = |\Omega_{\alpha}^{max}(t)|$. Similarity ratios do not change with time and they are $\lambda_1^3(t) = 0.2$, $\lambda_2^3(t) = 0.6$, $\lambda_3^3(t) = 1$, $\lambda_4^3(t) = 0.9$, and $\lambda_5^3(t) = 0.4$. All obtained experiences are uniformly distributed with mean equal to 0.5. Services have constant awareness, $k_1^w(t) = 0$, $k_2^w(t) = 0$, $k_3^w(t) = 1$, $k_4^w(t) = 1$, $k_5^w(t) = 0$. The aging factor and the service reputation are constant, $\tau(t) = 0.7$ and $r_3(t) = 0.2$.

6.5.2.2. Trust Assessment. Each entity may have different trust assessment approach. For instance, entities may have partial trust metrics that are about some properties of a service and total trust metrics that consider all properties of a service. On the other hand, the trust assessment depends on the context so each trust metric may have different computation models. Since our aim is to show effects of similar security experiences from many services on a trust assessment, we focus on the trust assessment based on similar security experiences and use the general trust assessment metric $tr_i^a(t)$. Therefore, we consider only an ultimate experience and the quality of the ultimate experience as in Equation 6.10 for a trust assessment.

According to Equation 6.9, the quality of an ultimate experience depends on the awareness of each similar service and the service reputation. The awareness of a service and the service reputation are constant in this case study. Therefore, quality of ultimate experience $k_3(t)$ is constant and Equation 6.10 is as Equation 6.11. Note that if we apply the facts, quality of ultimate experience k_3 is 0.6.

$$tr_i^a(t) = \epsilon_i(t)\, k_i. \tag{6.11}$$

Table 6.1 contains the first ten seconds of the trust assessment results about the security system of service ω_3 for entity α shown in Figure 6.11. It is obvious that ultimate experience $\epsilon_3(t)$ depends on the number of services. Because ultimate experience $\epsilon_3(t)$ is dynamic in time, assessed trust $tr_3^a(t)$ changes proportional to ultimate

experience $\epsilon_3(t)$. On the other hand, if entity α does not have security experiences about the security system of service ω_3, the entity does not trust that service, which means the initial trust in entity α is zero in this model. Therefore, entity α has to obtain at least one security experience either from the security system of service ω_3 or the security system of a similar service to have a trust value greater than zero.

6.5.2.3. Performance Evaluation. We simulated the scenario to carry out the trust assessment about the security system of service ω_3. The performance evaluation of the trust assessment was accomplished according to different number of services.

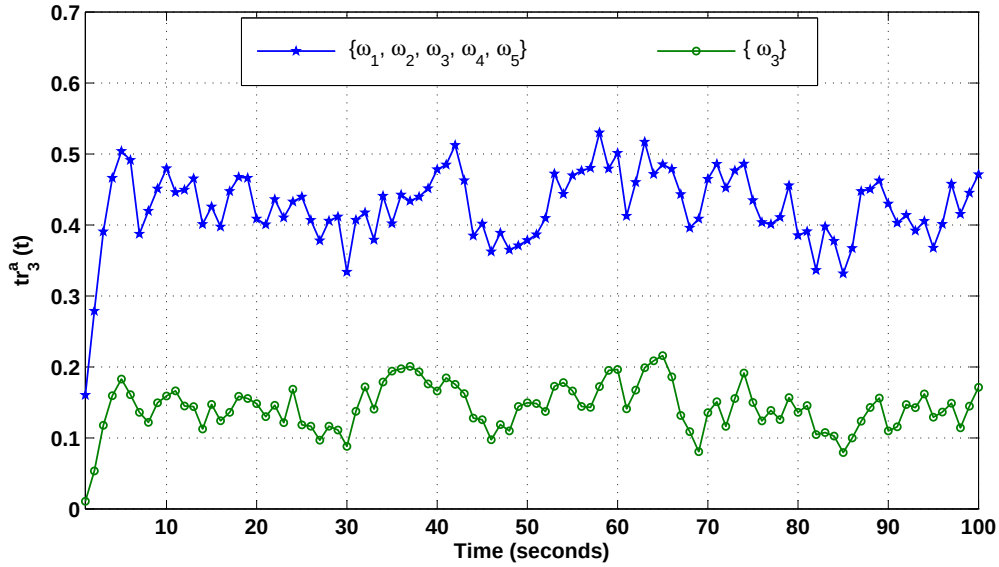

Figure 6.12. Trust assessment results for $N = 1$ and $N = 5$.

There are two simulation results shown in Figure 6.12 that shows the trust assessment results about the security system of service ω_3. In first simulation, the entity obtains security experiences only from service ω_3. In second simulation, entity α obtains security experiences from all services. The simulation results are consistent with the results in Table 6.1. Moreover, the simulation results show the advantage of multiple services over one service to assess the trust. Specifically, the amount of security experience is the most significant factor for a trust assessment. For example, if $N = 1$, the average value of assessed trust tr_3^a is 0.1402, whereas, if $N = 5$, the average value of assessed trust tr_3^a is 0.4326.

Table 6.1. Ultimate experience $\epsilon_3(t)$ and assessed trust $tr_3^a(t)$ for $N = 1$ and $N = 5$ with aging factor $\tau = 0.7$.

t	$\epsilon_1^r(t)$	$\epsilon_2^r(t)$	$\epsilon_3^r(t)$	$\epsilon_4^r(t)$	$\epsilon_5^r(t)$	$N=1$		$N=5$	
						$\epsilon_3(t)$	$tr_3^a(t)$	$\epsilon_3(t)$	$tr_3^a(t)$
0	0	0	0	0	0	0	0	0	0
1	0.8147	0.9058	0.1270	0.9134	0.6324	0.0178	0.0107	0.2672	0.1603
2	0.0975	0.2785	0.5469	0.9575	0.9649	0.0890	0.0534	0.4644	0.2786
3	0.1576	0.9706	0.9572	0.4854	0.8003	0.1963	0.1178	0.6510	0.3906
4	0.1419	0.4218	0.9157	0.7922	0.9595	0.2656	0.1594	0.7768	0.4661
5	0.6557	0.0357	0.8491	0.9340	0.6787	0.3048	0.1829	0.8397	0.5038
6	0.7577	0.7431	0.3922	0.6555	0.1712	0.2683	0.1610	0.8185	0.4911
7	0.7060	0.0318	0.2769	0.0462	0.0971	0.2266	0.1359	0.6454	0.3873
8	0.8235	0.6948	0.3171	0.9502	0.0344	0.2030	0.1218	0.6993	0.4196
9	0.4387	0.3816	0.7655	0.7952	0.1869	0.2493	0.1496	0.7517	0.4510
10	0.4898	0.4456	0.6463	0.7094	0.7547	0.2650	0.1590	0.7994	0.4797

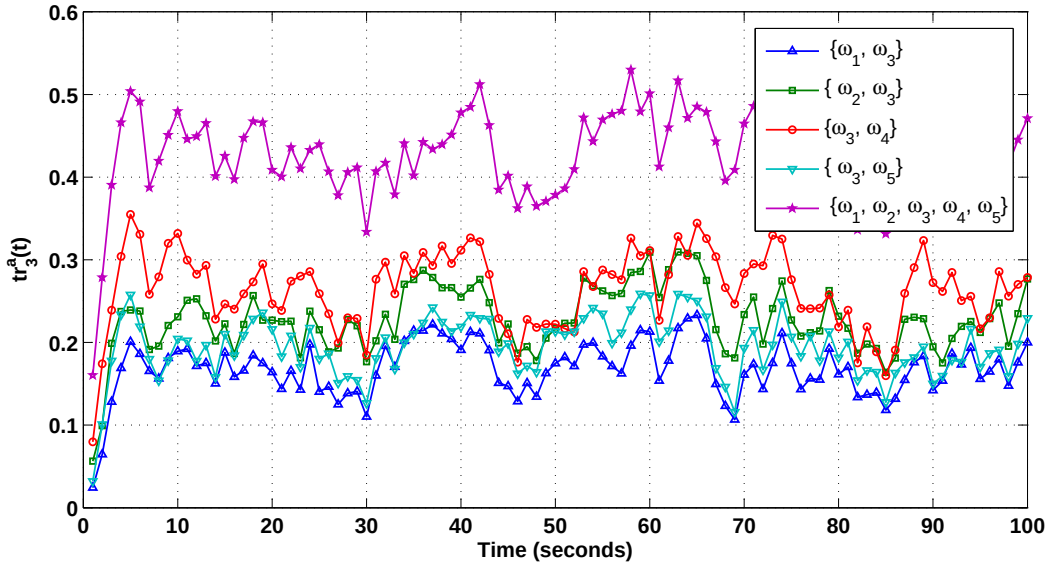

Figure 6.13. Trust assessment results for $N = 2$ and $N = 5$.

An entity may not know sufficient number of services that have similar security systems related to the security system of a specific service. For instance, if entity α knows one similar service in addition to service ω_3, the value of assessed trust $tr_3^a(t)$ will be more than if the entity gathers security experiences only from service ω_3. Figure 6.13 contains simulation results about the trust assessment for all possible $N = 2$ cases in addition to simulation results about the trust assessment for $N = 5$ case. Average values of assessed trust $tr_3^a(t)$ if $N = 2$ are 0.1683, 0.2243, 0.2647, and 0.1959 for $\{\omega_1, \omega_3\}$, $\{\omega_2, \omega_3\}$, $\{\omega_3, \omega_4\}$, and $\{\omega_3, \omega_5\}$ respectively. According to the simulation results, it is clear that the average value of $tr_3^a(t)$ for $N = 2$ is greater than the average value of assessed trust $tr_3^a(t)$ for $N = 1$, but the average value of assessed trust $tr_3^a(t)$ is less than for $N = 5$, as expected.

The average expected value of an assessed trust should increase when the number of services increases. We observe the expectation in the simulation results shown both in Figure 6.12 and in Figure 6.13.

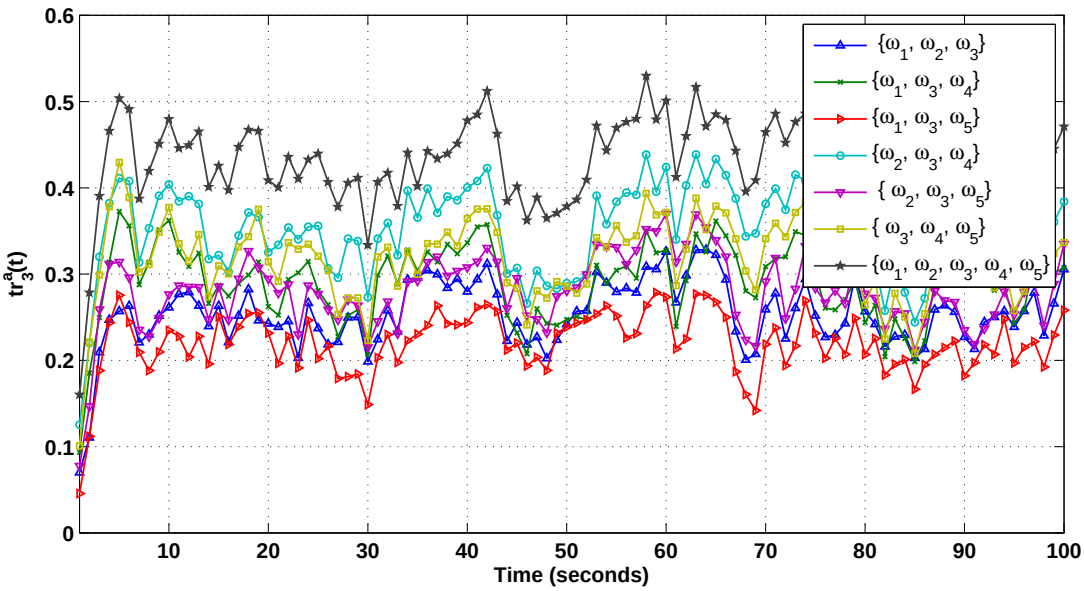

Figure 6.14. Trust assessment results for $N = 3$ and $N = 5$.

Different from the simulation results shown in Figure 6.12 and in Figure 6.13, some simulation results of assessed trust $tr_3^a(t)$ shown in Figure 6.14 and in Figure 6.15 contradict with the expected average assessed trust. For instance, Figure 6.14 and Figure 6.15 contain all possible simulation results related to $N = 3$ and $N = 4$ cases

in sequence. The average value of assessed trust $tr_3^a(t)$ is 0.2647 for $\{\omega_3, \omega_4\}$ if $N = 2$, whereas, the average value of assessed trust $tr_3^a(t)$ is 0.2240 for $\{\omega_1, \omega_3, \omega_5\}$ if $N = 3$. When entity α gathers security experiences from $\{\omega_1, \omega_2, \omega_3, \omega_5\}$, the average value of assessed trust $tr_3^a(t)$ is 0.3081 that is smaller than 0.3208, where the entity interacts with $\{\omega_3, \omega_4, \omega_5\}$. Actually, the assessed trust depends on the number of services, but it also depends on the value of similarity ratios.

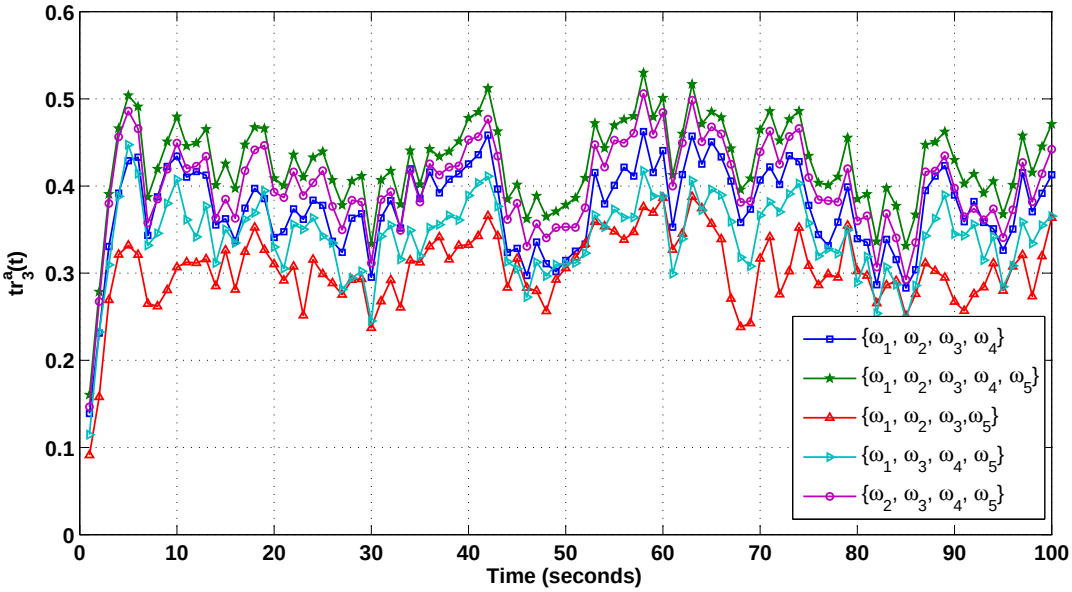

Figure 6.15. Trust assessment results for $N = 4$ and $N = 5$.

The effect of similarity ratios on the assessed trust is related to the values of similarity ratios. To show the effect, we use the *effective similarity ratio* that is computed with Equation 6.12. The value of the ultimate experience is high for larger values of effective similarity ratios if the same set of obtained experiences is used for computations, such as $efSim^5_{1,2,3,5} = 0.38$ and $efSim^5_{3,4,5} = 0.46$. We observe this for some instances of obtained experience because the average values of our obtained experiences are 0.5 for all simulations.

$$efSim^{maximumNumberOfServices}_{contributingServices} = \frac{\sum_{i=1}^{N} \lambda_i^{service}}{W_{entity}}. \tag{6.12}$$

The time needed to reach to a specific value of an assessed trust depends on the amount of security experiences so the value of an assessed trust implicitly depends on the number of services. If an entity obtains experiences from more than one service, it is expected that the time needed to reach a certain value of the assessed trust is less than the time needed in one service case. The simulation results in Table 6.2 verify this assertion. For example, entity α needs 37 seconds to compute assessed trust $tr_3^a(t) = 0.2$ if the entity gathers experiences only from service ω_3. On the other hand, two seconds are enough to compute assessed trust $tr_3^a(t) = 0.2$ in $N = 5$ case.

Table 6.2. The effect of similar security experiences on the performance for a trust assessment.

tr_3^a	0.1	0.2	0.3	0.4	0.5	0.6
$\{\omega_3\}$	3	37	-	-	-	-
$\{\omega_1,\omega_3\}$	3	5	-	-	-	-
$\{\omega_2,\omega_3\}$	3	4	60	-	-	-
$\{\omega_3,\omega_4\}$	2	3	4	4721	-	-
$\{\omega_3,\omega_5\}$	2	4	1940	-	-	-
$\{\omega_1,\omega_2,\omega_3\}$	2	3	36	-	-	-
$\{\omega_1,\omega_3,\omega_4\}$	2	3	4	262	-	-
$\{\omega_1,\omega_3,\omega_5\}$	2	4	122	-	-	-
$\{\omega_2,\omega_3,\omega_4\}$	1	2	3	5	1943	-
$\{\omega_2,\omega_3,\omega_5\}$	2	3	4	1940	-	-
$\{\omega_3,\omega_4,\omega_5\}$	1	2	4	5	-	-
$\{\omega_1,\omega_2,\omega_3,\omega_4\}$	1	2	3	5	262	-
$\{\omega_1,\omega_2,\omega_3,\omega_5\}$	2	3	4	122	-	-
$\{\omega_1,\omega_3,\omega_4,\omega_5\}$	1	2	3	5	3449	-
$\{\omega_2,\omega_3,\omega_4,\omega_5\}$	1	2	3	4	58	-
$\{\omega_1,\omega_2,\omega_3,\omega_4,\omega_5\}$	1	2	3	4	5	2108

Table 6.2 contains exceptional cases that contradict with the time assertion mentioned above. In some cases, an entity may have greater values of the effective similarity

ratio with less number of services. Therefore, the entity needs less time to compute a specific value of an assessed trust with less number of services as in some cases shown in Table 6.2.

This case study shows that having more services with the highest effective similarity ratio improves the performance of a trust assessment.

6.5.3. Case Study 3: Security Interoperability Decision Making for a User Accessing Multiple Services

The setting in this case study is the same as in Case Study 2. In Case Study 2, we analyzed the proposed trust assessment model. On the other hand, this case study explains how a security interoperability decision is made between an entity and a service by using security experiences from similar services so the security interoperability among services are established. Specifically, the agent that runs on the professor's mobile phone makes a security interoperability decision about the security system of service ω_3 based on trust assessment.

An entity may have no information about the security system of a service or the entity may have limited information about the security system of the service. If an entity uses conventional security approaches, the entity should not attempt to get a service in these cases. In conventional security approaches, an entity should know the security system of a service. Moreover, the entity should be aware of every change in security systems of services to keep up secure interactions with these services, but this is impossible in many cases in open environments.

On the other hand, if the security system of a service has an adequate trust value in an entity, the entity may use the service. This fact implies more service options for entities. Therefore, an entity can make better security interoperability decisions.

We evaluated the performance of the trust based security interoperability decision making between entity α and service ω_3 under different minimum trust values. In this

case study, we used facts and some trust assessment results given in Case Study 2. Security interoperability decisions were made according to algorithm shown in Figure 6.4.5.

Note that quality of ultimate experience $k_3(t) = 0.6$ and is constant. Because assessed trust $tr_3^a(t)$ is the multiplication of ultimate experience $\epsilon_3(t)$ and quality of ultimate experience $k_3(t)$, assessed trust $tr_3^a(t)$ never exceeds both ultimate experience $\epsilon_3(t)$ and quality of ultimate experience $k_3(t)$. For this reason, we selected minimum trust $tr_3^m(t)$ to be at most 0.6.

Table 6.3. Performance results for security interoperability based on trust assessment as percentage.

tr_3^m	0.1	0.2	0.3	0.4	0.6
$\{\omega_3\}$	87.78	3.47	0	0	0
$\{\omega_1,\omega_3\}$	98.08	18.63	0	0	0
$\{\omega_2,\omega_3\}$	99.96	72.62	2.59	0	0
$\{\omega_3,\omega_4\}$	99.99	92.43	22.23	0.03	0
$\{\omega_3,\omega_5\}$	99.80	45.63	0.06	0	0
$\{\omega_1,\omega_2,\omega_3\}$	99.99	90.70	12.01	0	0
$\{\omega_1,\omega_3,\omega_4\}$	99.99	97.97	43.73	0.82	0
$\{\omega_1,\omega_3,\omega_5\}$	99.99	73.93	1.78	0	0
$\{\omega_2,\omega_3,\omega_4\}$	100	99.91	83.40	15.31	0
$\{\omega_2,\omega_3,\omega_5\}$	99.99	97.56	31.20	0.07	0
$\{\omega_3,\omega_4,\omega_5\}$	100	99.62	66.21	4.76	0
$\{\omega_1,\omega_2,\omega_3,\omega_4\}$	100	99.98	93.57	32.29	0
$\{\omega_1,\omega_2,\omega_3,\omega_5\}$	99.99	99.65	57.56	1.18	0
$\{\omega_1,\omega_3,\omega_4,\omega_5\}$	100	99.96	83.90	14.80	0
$\{\omega_2,\omega_3,\omega_4,\omega_5\}$	100	99.99	97.93	53.46	0
$\{\omega_1,\omega_2,\omega_3,\omega_4,\omega_5\}$	100	99.99	99.59	73.50	0.02

Table 6.3 contains performance results of security interoperability decision mak-

ing based on the trust assessment between entity α and service ω_3 for five different values of minimum trust $tr_3^m(t)$. Generally, the possibility of a satisfactory security interoperability decision increases when the number of similar services increases. For example, when entity α considers only security experiences from service service service ω_3 with minimum trust $tr_3^m(t) = 0.1$, the entity has 87.78% security interoperability satisfaction for future interactions. Moreover, if the entity has minimum trust $tr_3^m(t) = 0.2$, the security interoperability satisfaction is only 3.47%. In contrast, if entity α considers experiences from five services for minimum trust $tr_3^m(t) = 0.1$ and minimum trust $tr_3^m(t) = 0.2$, the entity has 100% and 99.99% security interoperability satisfactions respectively. The results show that entity α has 12.22% and 96.52% performance gains for minimum trust $tr_3^m(t) = 0.1$ and $tr_3^m(t) = 0.2$ if the entity considers experiences from five services. Additionally, the performance improvement is obvious when N increases for all values of minimum trust $tr_3^m(t)$ as shown in Table 6.3.

On the other hand, obtaining experinces from more services gives opportunity to an entity to select more larger values of the minimum trust. Therefore, an entity can have more secure interactions with services. For instance, if entity α obtains security experiences only from service ω_3, the entity can select at most minimum trust $tr_3^m(t) = 0.2$. Otherwise, the entity will never have a satisfactory decision for interactions with service ω_3. However, if entity α takes into account experiences from all other services, it can select at most minimum trust $tr_3^m(t) = 0.6$. The results show that obtaining experiences from more services not only improves performance, but also ensures more secure interactions between an entity and a service.

The performance of security interoperability decisions depends also on the effective similarity ratio. Therefore, it is important for an entity to determine set $\Omega_\alpha^{max}(t)$. For example, if entity α has $N = 3$ and wants to interact with service ω_3, Table 6.3 shows the percentage of possible constructions of the set with $N = 3$. If $\Omega_\alpha^{max}(t) = \{\omega_1, \omega_3, \omega_5\}$, then the effective similarity ratio is 0.32 and the percentage of a satisfactory security interoperability decision with minimum trust $tr_3^m(t) = 0.3$ is 1.78%. In contrast, if $\Omega_\alpha^{max}(t) = \{\omega_2, \omega_3, \omega_4\}$, then the effective similarity ratio is 0.5 and the percentage of a satisfactory security interoperability decision is 83.40%

for minimum trust $tr_3^m(t) = 0.3$. The examples show the significance of the effective similarity ratio on the interoperability decision performance.

In general, the satisfaction of a security interoperability decision depends on minimum trust and assessed trust. Moreover, the satisfaction may be defined as a probability depending on minimum trust and assessed trust. For instance, for a number of discrete time periods, $1, \ldots, D$ an entity computes the satisfaction of a security interoperability decision based on the trust assessment related to the security system of a service in variables $t_1, \ldots, t_D$. Satisfaction and not satisfaction may be represented with 1 and 0 respectively. Let H_d stands for the hypothesis that: *t_i is 1 with probability d for all i.* This may be expressed probabilistically, $p(t_i|H_d) = d^{t_i}(1-d)^{1-t_i}$. The probability representation is a standard Bernoulli trial distribution.

Beta distribution may be convenient to analyze the satisfaction of a security interoperability decision because the beta distribution is the probability of success in Bernoulli distribution. The beta distribution is defined on the interval $(0, 1)$ with two parameters, which are generally denoted by entity α and β. Based on estimation of the parameters from real satisfaction instances, the distribution of satisfactions related to a specific service from the entity point of view may be found.

In this case study, we show that obtained security experiences from many services provide better security interoperability performance. Moreover, the amount of similar services increases the probability of satisfactory security interoperability decisions. Therefore, it is significant to consider services with similar security systems to ensure service convergence in networks.

6.6. Conclusion

Recently, networks have converged to ubiquitous networks that enable services to be accessible anywhere and anytime with different devices and different physical access networks. Ubiquitous networks necessitate convergence of services, where new research issues emerge, such as security and trust issues. In this chapter, we studied

the issue of security interoperability between services and entities, based on trust in ubiquitous networks to ensure service convergence. Trust computations are carried on entities by considering security experiences obtained from services that have similar security systems with the security system of a specific service.

We proposed a crust model to obtain security experiences about the security system of a specific service from many services for trust computations. In our model, security experiences from different services are aggregated based on similarity ratios and a similarity ratio determines the contribution of security experiences from a service to the computation of the trust. An entity computes similarity ratios about security systems of services by considering the security system of the specific service and its internal facts, such as utility relations with the service.

In the model, two types of trust metrics were proposed, namely the assessed trust and the minimum trust. The assessed trust represents the trust assessed by an entity about the security system of a service and is computed according to an ultimate experience and the quality of the ultimate experience. Each entity can determine a minimum trust value according to its needs. Furthermore, we proposed an algorithm for the security interoperability decision making based on the assessed trust and the minimum trust.

The chapter also contains the analysis of the proposed model. Three case studies were presented to demonstrate the advantages of our proposed model. Performance analyses by simulations were carried out for each case study. In the first case study, we concentrated on effects of some parameters for the computation of trust, such as the effects of an aging factor. In the second one, we explained and evaluated the computation of the assessed trust. Finally in the third case study, we showed the advantage of the security interoperability decision making based on the trust assessment. Performance analyses show that obtaining security experiences from many services related to the security system of a specific service provides better trust assessment results than obtaining security experiences from the specific service only. Moreover, our new algorithm provides high performance security interoperability decision making based on

trust. Therefore, we had better service convergence in ubiquitous networks.

7. CONCLUSION AND FUTURE WORK

The openness of services oriented environments has been increasing and trust has become a significant issue in that environments. Entities in such environments have different security needs from security systems of services. Assessing the trust of the security systems is a precondition to make trust based decisions for future interactions. Moreover, assessing the trust based on the needs of a specific entity necessitates specific models and precise representations of trust. This thesis presented our research into modeling and representation of trust related to security in emerging service oriented environments. In this chapter, we conclude the dissertation by summarizing our main contributions and then pointing out some future work.

7.1. Summary of Contributions

We began with an overview of research related to trust in computer science and we were focused on trust research related to security. Our goal was to present existing ideas and work to show the ways to design trust models and representations of trust in many contexts. Chapter 2 presents the relationship between trust and security, a systematic examination of trust, trust modeling and some applications of trust, and some open research problems related to trust. In this chapter, we described some of current research that exemplifies solutions in order to produce an applicable trust assessment system.

Trust has many definitions and various models that depend on their contexts, where the trust models do not comply with each other. To date, modeling of trust has suffered from the lack of a common trust model that can be applied to all entities and services in open environments. Actually, a unique trust model may not be applicable in an open environment because each entity may have different needs from security system of a service. Therefore, an entity should have trust models that reflect its needs from the security system of a service. An entity should also have a common trust model to be able to interact with other entities related to trust computations. Briefly, an

entity should have two types of trust models for trust computations related to security systems of services in open environments. The first type of trust models should be common to all entities whereas the second type of trust models can be specific to an entity.

We proposed a hybrid modeling approach to represent trust of security systems in open environments based on needs of entities, namely Core-Crust Modeling Approach (CCMA). CCMA contains two types of models. First type model is the core model that represents common requirements of entities from security systems of services related to trust computations. The other type model is a crust model that represents specific needs of the entity from the security systems of service in open environments. An entity may have more than one crust model but it has only one core model. The hybrid approach is suitable to the openness property of emerging open environments so that new trust models can be added and interoperability among the models can be ensured. More significantly, an entity can compute trust based on its special needs according to CCMA.

In CCMA, a crust model may be modified very frequently and entities may have different crust models in an open environment. On the other hand, all entities have same core model for trust computations. Therefore, a crust model is designed according to the core model. This means that a crust model and its related core model have to be connected. The connection is achieved with parameters of a core model and a crust model. Particularly, a crust model shares at least one parameter with the core model or with another crust model that has connection with the core model.

In this thesis, all crust models are directly connected to the core model. Moreover, two crust models are also connected with each other. Specifically, the core trust model was designed to assess the trust of a security system. The trust assessment process requires information for computations. One crust model extracts security evaluation information from security system of services. Another crust model obtains security evaluation information from other entities and services. The last crust model is more specific one which goal is to improve obtained security evaluation information from

many services related to the security system of a specific service.

We proposed our core model, namely Trust Core Model in Chapter 3. The core model is for assessing the trust of the security system of a service from entity point of view. The model contains architecture of the trust assessment system of an entity. The architecture can be applied to any entity in an open environment. In the core model, we also proposed novel trust metrics to assess trust of a security system. The trust assessment metrics are computed according to information obtained from three main sources. In particular, an entity may assess the trust of a security property of a security system by using partial metrics and it may use total metrics to assess trust of all properties of the security system. Moreover, each entity can use its own crust model(s) with this core model to compute the metrics according to its own needs. In this Chapter, we provided a Hospital Online Appointment Service as a case study with many simulations to show the applicability of our core model.

Next, we concentrated to obtain information for trust computations. To accomplish this task, we presented Trust Extraction Crust Model to extract trust information from the security system of a service based on specific needs of an entity. In the crust model, we formally represented security policies and security systems to extract trust information. The formal representation ensures an entity to compute both partial metrics and total metrics. We applied the curst model to Management Service of Patients' Records of a Dental Clinic as a case study. We analyzed the case study experimentally via simulations. The experimental evaluation showed that the crust model provides information for trust computations. The results show that the crust model can be easily adapted to our core model therefore an entity may use a specific model for extracting trust information according to our CCMA.

In our core model, there are three main sources of information for trust computations. In Chapter 4, we presented a crust model to extract information from the security system of a service for trust computations. In Chapter 5, we proposed a boarder formal model to obtain information for trust computations as a new crust model, namely Information Flow Crust Model. We called security evaluation informa-

tion to such information in the model. In this chapter, trust assessment is carried out according to flow of security evaluation information from many entities and services to a specific entity. On the other hand, we introduced subjective factors, such as similarity ratio, risk factor, accuracy factor, to obtain security evaluation information for trust computations. The subjective factors reflect behaviors of each individual entity during flow of security evaluation information. Chapter 5 contains a case study, Online Hotel Reservation Service, which was used to demonstrate advantages of our crust model for flow of security evaluation information and trust assessment. The case study was analyzed with simulations. The analysis results show that the crust model can improve the amount of information for trust assessments. Finally, the crust model is compatible with our core model and CCMA.

We presented a more specific crust model, manely Interoperability Crust Model in Chapter 6. The model was designed to obtain security experiences related to a specific service in a faster manner. Moreover, our another aim was to increase the accuracy of trust computations by using more security experiences related to a service. In particular, Chapter 6 describes a formal model to make trust based security interoperability decisions for service convergence according to needs of a specific entity in networks. In this model, security systems of services are similar. A similarity ratio determines the amount of contribution of the obtained experience from a service to compute ultimate experience. We also provided the computation of the similarity ratio. Furthermore, trust metrics are computed according to security information obtained from services that have similar security systems. We then made security interoperability decisions based on values of the trust metrics. We supported the crust model with case studies. Performance analyses had been carried out by simulations for each case study. Performance analyses show that obtaining security experiences from many services related to the security system of a specific service provides better trust assessment results than obtaining security experiences from single service. Therefore, an entity can make better security interoperability decisions based on trust with the crust model. This crust model also show that entities can have their own crust models in our CCMA.

7.2. Future Work

The work contained in this thesis is about modeling and formal representation of trust in relation to security based on the needs of a specific entity. There are many open problems in the field of trust research. We have some suggestions to improve the research related to this study. Our suggestions are as follows.

- In this thesis, we proposed a core-crust modeling approach and we presented models both for core and for crust in security context for emerging open environments. However, trust has been investigating in many contexts. Hence, CCMA for trust modeling may be applied to other contexts, such as routing in networks.

- We proposed only one core model for trust assessment of the security system of a service based on needs of an entity. The aim of the core model is to show applicability of CCMA. More detailed core models may be designed by considering specific properties of entities and services in emerging open environments.

- We introduced three crust models to show the applicability of CCMA. More specific crust models can be designed according to tasks of entities. Moreover, the crust models can be used for trust based decision making and the decisions can be used for many purposes as described in Chapter 6. Therefore, designing crust models according to specific needs of entities necessitates huge effort.

- Since CCMA is extendible, additional crust models can be envisioned and added to CCMA.

- In this thesis, we introduced many subjective factors. However, we proposed only a method for computation of the similarity ratio. However, each entity may have different methods to determine the subjective factors. Moreover, entities needs formal model to compute each subjective factor. Therefore, formal models for computations of the subjective factors are needed.

- We provided case studies to show the applicability of our proposed models. Much complicated case studies may be used to analyze the proposed models. In addition, implementations of the proposed models on entities are also future work.

REFERENCES

1. Massa, P., *Trust in E-services: Technologies, Practices and Challenges*, chap. A Survey of Trust Use and Modeling in Real Online Systems, pp. 51–83, Idea Group Inc., Hershey, PA, USA, 2007.

2. Chivers, H., *Security and Systems Engineering*, Technical Report, Department of Computer Engineering, University of York, Heslington, York, UK, 1994.

3. Kagal, L., T. Finin and A. Joshi, "Trust-Based Security in Pervasive Computing Environments", *IEEE Computer*, Vol. 34, No. 12, pp. 154–157, 2001.

4. Grandison, T. and M. Sloman, "A Survey of Trust in Internet Applications", *IEEE Communications Survey*, Vol. 3, No. 4, pp. 2–16, 2000.

5. Krukow, K., *Towards a Theory of Trust for the Global Ubiquitous Computer*, Ph.D. Thesis, University of Aarhus, 2006.

6. Jin, Y., J. Zhang and X. Zheng, "Specification and Runtime Enforcement of Security Policies", *IFIP International Conference on Network and Parallel Computing Workshops, NPC '07*, Washington, DC, USA, 18-21 September, IEEE Computer Society, pp. 244–249, 2007,.

7. Li, J., J. Huai and C. Hu, "PEACE-VO: A Secure Policy-Enabled Collaboration Framework for Virtual Organizations", *26th IEEE International Symposium on Reliable Distributed Systems, SRDS '07*, Washington, DC, USA, 10-12 October, IEEE Computer Society, pp. 199–208, 2007.

8. Patz, G., M. Condell, R. Krishnan and L. Sanchez, "Multidimensional security policy management for dynamic coalitions", *DARPA Information Survivability Conference & Exposition II, DISCEX '01*, Anaheim, CA, USA, 13-14 June, Internet Society, pp. 41–54, 2001.

9. Weise, J. and C. R. Martin, *Developing a Security Policy*, Technical Report, Sun Microsystems, Inc., 901 San Antonio Road, Palo Alto, CA 94303 USA, 2001.

10. Sandhu, R. S., X. Zhang, K. Ranganathan and M. J. Covington, "Client-side access control enforcement using trusted computing and PEI models", *Journal of High Speed Networks*, Vol. 15, No. 3, pp. 229–245, 2006.

11. Wang, F., "Formal verification of timed systems: a survey and perspective", *Proceedings of the IEEE*, 19 July, IEEE, Vol. 92, No. 8, pp. 1283 – 1305, 2004.

12. Reinbacher, T., M. Kramer, M. Horauer and B. Schlich, "Challenges in embedded model checking a simulator for the [mc]square model checker", *International Symposium on Industrial Embedded Systems, SIES 2008*, La Grande Motte, France, 11-13 June, IEEE, pp. 245-248, 2008.

13. Devaraj, G., M. Heimdahl and D. Liang, "Coverage-directed test generation with model checkers: challenges and opportunities", *29th Annual International Computer Software and Applications Conference, COMPSAC 2005*, Edinburgh, Scotland, UK, 25-28 July, IEEE Computer Society, pp. 455-462, 2005.

14. Jha, S., N. Li, M. Tripunitara, Q. Wang and W. H. Winsborough, "Toward Formal Verification of Role-Based Access Control Policies", *IEEE Transactions on Dependable and Secure Computing*, Vol. 5, No. 4, pp. 1–14, 2008.

15. Katoen, J., "Perspectives in Probabilistic Verification", *2nd IFIP/IEEE International Symposium on Theoretical Aspects of Software Engineering, TASE '08*, Nanjing, China, 17-19 June, IEEE Computer Society, pp. 3-10, 2008.

16. D'Silva, V., D. Kroening and G. Weissenbacher, "A Survey of Automated Techniques for Formal Software Verification", *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, Vol. 27, No. 7, pp. 1165–1178, 2008.

17. Andert, D., R. Wakefield and J. Weise, *Trust Modeling for Security Architecture*

Development, Technical Report, Sun Microsystems, Inc., Santa Clara, CA, USA, 2002.

18. Sun, Z., Y. L. Han and K. J. R. Liu, "Defense of trust management vulnerabilities in distributed networks", *IEEE Communications Magazine*, Vol. 46, No. 2, pp. 112–119, 2008.

19. Yan, Z., *Trust Management for Mobile Computing Platforms*, Ph.D. Thesis, Helsinki University of Technology, 2007.

20. Gollmann, D., "Why Trust is Bad for Security", *Electronic Notes in Theoretical Computer Science*, Vol. 157, No. 3, pp. 3–9, 2006.

21. Bahtiyar, Ş. and M. U. Çağlayan, "Trust Assessment of a Security System from the Entity Point of View", *Decision Support Systems*, submitted.

22. Bahtiyar, Ş. and M. U. Çağlayan, "Trust Computations Based on Similarity of Security Systems", *Computer Standards and Interfaces*, submitted.

23. Bahtiyar, Ş., M. Cihan and M. U. Çağlayan, "An architectural approach for assessing system trust based on security policy specifications and security mechanisms", *Proceedings of the 2nd international conference on Security of information and networks SIN '09/*, Gazimagusa, North Cyprus, 6-10 October, ACM, pp. 71–74, 2009.

24. Bahtiyar, Ş. and M. U. Çağlayan, "Extracting trust information from security system of a service", *Journal of Network and Computer Applications*, Vol. 35, No. 1, pp. 480–490, 2012.

25. Bahtiyar, Ş. and M. U. Çağlayan, "Trust Assessment Based on Flow of Security Evaluation Information on Entities", *Concurrency and Computation: Practice and Experience*, submitted.

26. Bahtiyar, Ş. M. Cihan and M. U. Çağlayan, "A Model of Security Information

Flow on Entities for Trust Computation", *Computer and Information Technology, International Conference on*, Bradford, West Yorkshire, UK, 29 June - 1 July, IEEE Computer Society, pp. 803–808, 2010.

27. Bahtiyar, Ş., M. Cihan and M. U. Çağlayan, "Security Information Propagation on Entities for Trust Assessment", *4th IFIP WG 11.11 International Conference on Trust Management*, Morioka, Iwate, Japan, 14-18 June, pp. 127–134, 2010.

28. Kovač, D. and D. Trček, "Qualitative trust modeling in SOA", *Journal of Systems Architecture*, Vol. 55, No. 4, pp. 255–263, 2009.

29. Jøsang, A., "Trust and Reputation Systems", *In A. Aldini and R. Gorrieri (Eds.), Foundations of Security Analysis and Design IV, FOSAD 2006/2007 Tutorial Lectures. Springer LNCS 4677*, Vol. 4677, 2007.

30. Rasmusson, L. and S. Jansson, "Simulated social control for secure Internet commerce", *Proceedings of the 1996 workshop on New security paradigms , NSPW '96 /*, Lake Arrowhead, California, USA, 17 - 20 September, ACM, pp. 18–25, 1996.

31. Hexmoor, H., S. Wilson and S. Bhattaram, "A theoretical inter-organizational trust-based security model", *The Knowledge Engineering Review*, Vol. 21, No. 2, pp. 127–161, 2006.

32. Nadalin, A., C. Kaler, R. Monzillo and P. Hallam-Baker, "Web Services Security: SOAP Message Security 1.1 (WS-Security 2004)", 2006, docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-os-SOAPMessageSecurity.pdf, 2009.

33. Della-Libera, G., M. Gudgin, P. Hallam-Baker, M. Hondo, H. Granqvist, C. Kaler, H. Maruyama, M. McIntosh, A. Nadalin, N. Nagaratnam, R. Philpott, H. Prafullchandra, J. Shewchuk, D. Walter and R. Zolfonoon, "Web Services Security Policy Language(WS-SecurityPolicy)", 2005, http://specs.xmlsoap.org/ws/2005/07/securitypolicy/ws-securitypolicy.pdf,

2009.

34. Nadalin, A., M. Goodner, M. Gudgin, A. Barbir and H. Granqvist, "WS-Trust 1.3", 2007, http://docs.oasis-open.org/ws-sx/ws-trust/200512/ws-trust-1.3-os.html, 2009.

35. Tan, L., S. yi Zhang and H. yun Wang, "A Group Trust Mechanism Based on User Behaviors and Policy Trust in Trustworthy Routing Systems", *Proceedings of the IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application PACIIA '08/*, Wuhan, China, December, 19-20 December, IEEE Computer Society, pp. 548-552, 2008.

36. Bertocco, C. and C. Ferrari, "Context-Dependent Reputation Management for Soft Security in Multi Agent Systems", *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology WI-IAT '08/*,Sydney, NSW, 9-12 December, IEEE Computer Society, Vol. 3, pp. 77–81, 2008.

37. Fullam, K. K., J. Park and K. S. Barber, "Trust-driven Information Acquisition for Secure and Quality Decision-Making", *International Conference on Integration of Knowledge Intensive Multi-Agent Systems*, Waltham, MA, USA, April, IEEE, pp. 303 – 310, 2005.

38. Zhu, Z. and G. Wu, "Statistical Analysis of Trust Overlay Network", *Proceedings of the Second International Conference on Future Information Technology and Management Engineering ,FITME '09 /*, Sanya, China, 13-14 December, IEEE Computer Society, pp. 592–595, 2009.

39. Raya, M., P. Papadimitratos, V. Gligor and J. Hubaux, "On Data-Centric Trust Establishment in Ephemeral Ad Hoc Networks", *The 27th Conference on Computer Communications INFOCOM/*, Phoenix, AZ, USA, 13-18 April, IEEE, pp. 1238 - 1246, 2008.

40. Bertino, E., L. R. Khan, R. Sandhu and B. Thuraisingham, "Secure Knowledge Management: Confidentiality, Trust, and Privacy", *IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans*, Vol. 36, No. 3, pp. 429–438, 2006.

41. Gambetta, D., "Can We Trust Trust?", *Trust: Making and Breaking Cooperative Relations*, pp. 213–237, Basil Blackwell, 1988.

42. Hoven, D. J. D., "Computer Ethics and Moral Methodology", *The Methaphilosophy Foundation and Blackwell Publishers Ltd.*, Vol. 28, No. 3, pp. 234–247, 1997.

43. Deutsch, M., "Trust and Suspicion", *The Journal of Conflict Resolution*, Vol. 2, No. 4, pp. 265–279, 1958.

44. Misztal, B., *Trust in Modern Societies: The Search for the Bases of Social Order*, Polity Press, Malden, MA, USA, 1996.

45. Mayer, R. C., J. H. Davis and F. D. Schoorman, "An Integrative Model of Organizational Trust", *The Academy of Management Review*, Vol. 20, No. 3, pp. 709–734, 1995.

46. Jøsang, A., R. Ismail and C. Boyd, "A Survey of Trust and Reputation Systems for Online Service Provision", *Decision Support Systems*, Vol. 43, No. 2, pp. 618–644, 2007.

47. Weeks, S., "Understanding trust management systems", *IEEE Symposium on Security and Privacy, S&P 2001*, Oakland, California, USA, 13-16 May, IEEE Computer Society, pp. 94-105, 2001.

48. Kuter, U. and J. Golbeck, "SUNNY: A New Algorithm for Trust Inference in Social Networks Using Probabilistic Confidence Models", *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence*, Vancouver, British Columbia,

Canada, 22-26 July, AAAI Press, pp. 1377–1382, 2007.

49. Ryutov, T., "A Socio-cognitive Approach to Modeling Policies in Open Environments", *Eighth IEEE International Workshop on Policies for Distributed Systems and Networks, POLICY '07*, Bologna, Italy, 13-15 June, IEEE Computer Society, pp. 29–38, 2007.

50. Jøsang, A., "Prospectives for Online Trust Management", May 2008, draft, http://folk.uio.no/josang/papers/Jos2007-oltrustman.pdf.

51. Falcone, R. and C. Castelfranchi, *Trust and deception in virtual societies*, chap. Social trust: a cognitive approach, pp. 55–90, Kluwer Academic Publishers, Dordrecht, Netherlands, 2001.

52. Mui, L., M. Mohtashemi and A. Halberstadt, "A Computational Model of Trust and Reputation for E-businesses", *Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS'02)*, Vol. 7 of *HICSS '02*, Hawaii, USA, 7-10 January, IEEE Computer Society, pp. 188, 2002.

53. Ajayi, O., R. Sinnott and A. Stell, "Trust Realisation in Multi-domain Collaborative Environments", *6th IEEE/ACIS International Conference on Computer and Information Science, ICIS 2007*, Melbourne, Qld, 11-13 July, IEEE, pp. 906 - 911, 2007.

54. Blaze, M., J. Feigenbaum and J. Lacy, "Decentralized trust management", *IEEE Symposium on Security and Privacy*, Oakland, CA, USA, May, IEEE Computer Society, pp. 164–173, 1996.

55. Trček, D., "A formal apparatus for modeling trust in computing environments", *Mathematical and Computer Modelling*, Vol. 49, No. 1-2, pp. 226–233, 2009.

56. Walter, T., L. Bussard, P. Robinson and R. Y., "Security and Trust Issues in Ubiquitous Environments – The Business-to-Employee Dimension", *Proceedings*

of the 2004 Symposium on Applications and the Internet-Workshops (SAINT 2004 Workshops), Tokyo, Japan, 26-30 January, IEEE, pp. 696 - 701, 2004.

57. Hussain, F., E. Chang and T. Dillon, "Comparative Analysis of Trust and Security", *IEEE International Conference on Service Operations and Logistics, and Informatics, SOLI '06*, Shanghai, China, 21-23 June, IEEE, pp. 1019 - 1023, 2006.

58. Olivieroa, F., L. Pelusoa and S. Romano, "REFACING: An autonomic approach to network security based on multidimensional trustworthiness", *Computer Networks*, Vol. 52, No. 14, pp. 2745–2763, 2008.

59. Barber, K. S., K. Fullam and J. Kim, *Trust, Reputation, and Security: Theories and Practice*, chap. Challenges for Trust, Fraud and Deception Research in Multi-agent Systems, pp. 167–174, Springer Berlin / Heidelberg, 2003.

60. Jøsang, A., R. Hayward and S. Pope, "Trust Network Analysis with Subjective Logic", *Proceedings of the 29th Australasian Computer Science Conference*,Hobart, Australia, 16-19 January, Australian Computer Society, pp.85–94, 2006.

61. Page, L., S. Brin, R. Motwani and T. Winograd, *The PageRank Citation Ranking: Bringing Order to the Web*, Technical Report, Stanford University, 1998.

62. Blaze, M., S. Kannan, I. Lee, O. Sokolsky, J. M. Smith, A. D. Keremytis and W. Lee, "Dynamic Trust Management", *IEEE Computer*, Vol. 42, No. 2, pp. 44–52, 2009.

63. Hoffman, L. J., K. Lawson-Jenkins and J. J. Blum, "Trust Beyond Security: An Expanded Trust Model", *Communications of the ACM*, Vol. 49, No. 7, pp. 94–101, 2006.

64. Djordjevic, I., S. Nair and T. Dimitrakos, "Virtualised Trusted Computing Plat-

form for Adaptive Security Enforcement of Web Services Interactions", *IEEE International Conference on Web Services, ICWS 2007*, Salt Lake City, Utah, USA, 9-13 July, IEEE Computer Society, pp. 615-622, 2007.

65. Herrmann, P., "Trust-Based Protection of Software Component Users and Designers", *1st International Conference on Trust Management*, Heraklion, Crete, Greece, May, Springer-Verlag, pp. 75–90, 2003.

66. Viega, J., T. Kohno and B. Potter, "Trust and Mistrust in Secure Applications", *Communications of the ACM*, Vol. 44, No. 2, pp. 31– 36, 2001.

67. Bichsel, P., S. Mller, F. Preiss, D. Sommer and M. Verdicchio, "Security and Trust through Electronic Social Networks-based Interactions", *In Workshop on Security and Privacy in Online Social Networking (SPOSN '09)*, Vancouver, BC, 29-31 August, IEEE Computer Society Press, pp. 1002–1007, 2009.

68. Fogel, J. and E. Nehmad, "Internet social network communities: Risk taking, trust, and privacy concerns", *Computers in Human Behavior*, Vol. 25, No. 1, pp. 153–160, 2009.

69. Squicciarini, A. C., E. Bertino, E. Ferrari and I. Ray, "Achieving Privacy in Trust Negotiations with an Ontology-Based Approach", *IEEE Transactions on Dependable and Secure Computing*, Vol. 3, No. 1, pp. 13–30, 2006.

70. Mármol, F. G. and G. M. Pérez, "Security threats scenarios in trust and reputation models for distributed systems", *Computers & Security*, Vol. 28, No. 7, pp. 545–556, 2009.

71. Kaufman, L. M., "Can a Trusted Environment Provide Security?", *IEEE Security&Privacy*, Vol. 8, No. 1, pp. 50–52, 2010.

72. Dimmock, N., A. Belokosztolszki, D. Eyers, J. Bacon and K. Moody, "Using trust and risk in role-based access control policies", *Proceedings of the ninth ACM*

symposium on Access control models and technologies SACMAT '04/, New York, NY, USA, 2-4 June, ACM, pp. 156–162, 2004.

73. Jøsang, A., "Subjective Logic", July 2008, draft, http://folk.uio.no/josang/papers/subjective_logic.pdf.

74. Ray, I., I. Ray and S. Chakraborty, "An interoperable context sensitive model of trust", *Journal of Intelligent Information Systems*, Vol. 32, No. 1, pp. 75–104, 2009.

75. Refsdal, A. and K. Stolen, "Extending UML sequence diagrams to model trust-dependent behavior with the aim to support risk analysis", *Electronic Notes in Theoretical Computer Science (ENTCS)*, Vol. 197, No. 2, pp. 15–29, 2008.

76. Bharadwaj, K. K. and M. Y. H. Al-Shamiri, "Fuzzy computational models for trust and reputation systems", *Elsevier Journal of Electronic Commerce Research and Applications*, Vol. 8, No. 1, pp. 37–47, 2009.

77. He, R.,G. Niu and J. Zhang, *CBTM: A Trust Model with Uncertainty Quantification and Reasoning for Pervasive Computing*, chap. Parallel and Distributed Processing and Applications, Lecture Notes in Computer Science, Springer Berlin / Heidelberg, pp. 541–552, 2005.

78. Shi, J., G. Bochmann and C. Adams, "A Trust Model with Statistical Foundation", *Second IFIP TC1 WG1.7 Workshop on Formal Aspects in Security and Trust, FAST*, Toulouse, France, 22-27 August, Springer, pp. 145-158, 2005.

79. Mezzetti, N., *A Socially Inspired Reputation Model*, Vol. 3093/2004 of *Lecture Notes in Computer Science*, chap. Public Key Infrastructure, Springer Berlin / Heidelberg, pp. 605 – 619, 2004.

80. Marsh, S. P., *Formalising Trust as a Computational Concept*, Ph.D. Thesis, University of Stirling, 1994.

81. Xiong, K. and H. Perros, "Trustworthy Web services provisioning for differentiated customer services", *Telecommunication Systems*, Vol. 39, No. 3-4, pp. 171185, 2008.

82. Skogsrud, H., H. R. Motahari-Nezhad, B. Benatallah and F. Casati, "Modeling Trust Negotiation for Web Services", *IEEE Computer*, Vol. 42, No. 2, pp. 54–61, 2009.

83. Patrick, A. S., "Building Trustworthy Software Agents", *IEEE Internet Computing*, Vol. 6, No. 6, pp. 46–53, 2002.

84. Ahamed, S. I. and M. Sharmin, "A trust-based secure service discovery (TSSD) model for pervasive computing", *Elsevier Journal of Computer Communications*, Vol. 31, No. 18, pp. 4281–4293, 2008.

85. Marti, S., *Trust and Reputation in Peer-to-Peer Networks*, Ph.D. Thesis, Stanford University, 2005.

86. Despotovic, Z. and K. Aberer, "P2P reputation management: probabilistic estimation vs. social networks", *Computer Networks*, Vol. 50, No. 4, pp. 485–500, 2006.

87. Ahamed, S. I., M. M. Haque, M. E. Hoque, F. Rahman and N. Talukder, "Design, analysis, and deployment of omnipresent Formal Trust Model (FTM) with trust bootstrapping for pervasive environments", *Journal of System and Software*, Vol. 83, No. 2, pp. 253–270, 2010.

88. Wang, Y., V. Cahill, E. Gray, C. Harris and L. Liao, "Bayesian Network Based Trust Management", *Third International Conference Autonomic and Trusted Computing, ATC'06*, Wuhan, China, 3-6 September, Springer, pp. 246–257, 2006.

89. Kong, W. C. and Y. T. C. Hung, "Modeling Initial and Repeat Online Trust in B2C E-Commerce", *Proceedings of the 39th Annual Hawaii International Confer-*

ence on System Sciences, Washington, DC, USA, 04-07 January, IEEE Computer Society, pp. 1–10, 2006.

90. Weng, J., Z. Shen, C. Miao, A. Goh and C. Leung, "Credibility: How Agents Can Handle Unfair Third-Party Testimonies in Computational Trust Models", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 22, No. 9, pp. 1286–1298, 2010.

91. Kamvar, S. D., M. T. Schlosser and H. Garcia-Molina, "The Eigentrust algorithm for reputation management in P2P networks", *Proceedings of the 12th international conference on World Wide Web*,Budapest, Hungary, 20-24 May, ACM, pp. 640–651, 2003.

92. Xiong, L. and L. Liu, "PeerTrust: Supporting Reputation-Based Trust for Peer-to-Peer Electronic Communities", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 16, No. 7, pp. 843–857, 2004.

93. Xiao-yong, L. and Z. Feng, "Developing P2P Trust Computing Mechanism based on Risk Evaluation Properties", *2nd International Conference on Advanced Computer Control (ICACC)*, Shenyang, 27-29 March, IEEE, pp. 409–413, 2010.

94. Ruizhong, D., T. Junfeng, W. Zixian and M. Xiaoxue, "A Trust Model of P2P Network Based on Reputation and Risk", *World Congress on Software Engineering, WCSE2009*, Xiamen, China, 19-21 May, IEEE, pp. 382–386, 2009.

95. Xiaonian, W., Z. Runlian, Z. Shengyuan and M. Chunbo, "Behavior Trust Computation Model Based on Risk Evaluation in the Grid Environment", *World Congress on Software Engineering, WCSE2009*, Xiamen, China, 19-21 May, IEEE, pp. 392–396, 2009.

96. Hussain, O. K., E. Chang, F. K. Hussain, T. S. Dillon and B. Soh, "Risk in Trusted Decentralized Communications", *Proceedings of the International Workshop on Privacy Data Management in Conjunction with 21st International Conference on*

Data Engineering (ICDE PDM 2005), Tokyo, Japan, 05-08 April, IEEE Computer Society, pp.1198 - 1198, 2005.

97. Gefen, D., V. S. Rao and N. Tractinsky, "The Conceptualization of Trust, Risk and Their Relationship in Electronic Commerce: The Need for Clarifications", *Proceedings of the 36th Annual Hawaii International Conference on System Sciences*, Hawaii, USA, 4-7 January, IEEE, pp. 1–10, 2003.

98. Lund, M., B. Solhaug and K. Stlen, "Evolution in Relation to Risk and Trust Management", *IEEE Computer*, Vol. 43, No. 5, pp. 49–55, 2010.

99. Solhaug, B., D. Elgesem and K. Stølen, "Why Trust is not Proportional to Risk", *Proceedings of the The Second International Conference on Availability, Reliability and Security(ARES'07)*, Vienna, Austria, 10-13 April, IEEE, pp. 11–18, 2007.

100. Zimmer, J. C., R. E. Arsal, M. Al-Marzouq and V. Grover, "Investigating online information disclosure: Effects of information relevance, trust and risk", *Elsevier Information&Management*, Vol. 47, No. 2, pp. 115–123, 2010.

101. Yu, W. and L. Jie, "Trust Risk of Enterprise Knowledge Sharing within Collaborative Commerce", *Chinese Control and Decision Conference (CCDC 2008)*, Yantai, Shandong, China, 2-4 July, IEEE, pp. 4500–4504, 2008.

102. Asnar, Y., P. Giorgini, F. Massacci and N. Zannone, "From Trust to Dependability through Risk Analysis", *Proceedings of the The Second International Conference on Availability, Reliability and Security (ARES'07)*, Vienna, Austria, 10-13 April, IEEE, pp. 19–26, 2007.

103. Quinn, K., D. Lewis and D. O 'Sullivan, "An analysis of accuracy experiments carried out over of a multi-faceted model of trust", *International Journal of Information Security*, Vol. 8, pp. 103–119, No. 2, 2009.

104. Heiskanen, A., M. Newman and M. Eklin, "Control, trust, power, and the dy-

namics of information system outsourcing relationships: A process study of contractual software development", *The Journal of Strategic Information Systems*, Vol. 17, pp. 268–286, No. 4, 2008.

105. Yu, B. and M. P. Singh, "A Social Mechanism of Reputation Management in Electronic Communities", *Proceedings of Fourth International Workshop on Cooperative Information Agents*, Vol. 1860 of *Lecture Notes In Computer Science*, pp. 154 – 165, 2000.

106. Jonker, C. M., J. J. P. Schalken, J. Theeuwes and J. Treur, "Human Experiments in Trust Dynamics", *iTrust*, Vol. 2995 of *Lecture Notes in Computer Science*, Springer, pp. 206–220, 2004.

107. AlZomai, M., B. AlFayyadh, A. Jøsang and A. McCullagh, "An Experimental Investigation of the Usability of Transaction Authorization in Online Bank Security Systems", *Proceedings of the Australasian Information Security Conference (AISC'08)*, Wollongong, Australia, January, Australian Computer Society, pp. 53–58, 2008.

108. Canfora, G., E. Costante, I. Pennino and C. A. Visaggio, "A three-layered model to implement data privacy policies", *Computer Standards & Interfaces*, Vol. 30, No. 6, pp. 398–409, 2008.

109. Biskup, J., J. Heilscher and S. Wortmann, "A Trust- and Property-based Access Control Model", *Electronic Notes in Theoretical Computer Science (ENTCS)*, Vol. 197, No. 2, pp. 169–177, 2008.

110. Malik, Z. and A. Bouguettaya, "Rater Credibility Assessment in Web Services Interactions", *World Wide Web*, Vol. 12, No. 1, pp. 3–25, 2009.

111. Ma, J. and M. Orgun, "Trust management and trust theory revision", *IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans*, Vol. 36, No. 3, pp. 451–460, 2006.

112. Theodorakopoulos, G. and J. S. Baras, "On Trust Models and Trust Evaluation Metrics for Ad Hoc Networks", *IEEE Journal on Selected Areas in Communications*, Vol. 24, No. 2, pp. 318–328, 2006.

113. Peng, S., W. Jia, G. Wang, J. Wu and M. Guo, "Trusted Routing Based on Dynamic Trust Mechanism in Mobile Ad-Hoc Networks", *IEICE Transactions on Information and Systems*, Vol. E93-D, No. 3, pp. 510–517, 2010.

114. TalebiFard, P., T. Wong and V. C. Leung, "Access and service convergence over the mobile internet A survey", *Computer Networks*, Vol. 54, No. 4, pp. 545–557, 2010.

115. Subashini, S. and V. Kavitha, "A survey on security issues in service delivery models of cloud computing", *Journal of Network and Computer Applications*, Vol. 34, No. 1, pp. 1–11, 2011.

116. Urbano, J., A. P. Rocha and E. Oliveira, "Trustworthiness Tendency Incremental Extraction Using Information Gain", *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Volume 02, WI-IAT '10*, Toronto, ON, 31 August - 3 September, IEEE Computer Society, pp. 411–414, 2010.

117. Jung, K. and Y. Lee, "Autonomic Trust Extraction for Trustworthy Service Discovery in Urban Computing", *Proceedings of the 2009 Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing, DASC '09*, Chengdu, 12-14 December, IEEE, pp. 502–507, 2009.

118. Zhang, H., J. Luo, F. Yan, M. Xu, F. He and J. Zhan, "A Practical Solution to Trusted Computing Platform Testing", *Third Asia-Pacific Trusted Infrastructure Technologies Conference, APTC '08*, Hubei, 14-17 October, IEEE, pp.79 - 87, 2008.

119. Lopez, J., R. Roman, I. Agudo and M. C. F. Gago, "Trust management systems

for wireless sensor networks: Best practices", *Computer Communications*, Vol. 33, No. 9, pp. 1086–1093, 2010.

120. Zhu, Y., Y. Li and Y. Ren, "Research on propagation of trust and distrust by means of co-citation", *6th International Conference on Service Systems and Service Management, ICSSSM '09*, Xiamen, 8-10 June, IEEE, pp. 943–948, 2009.

121. Jøsang, A., S. Pope and S. Marsh, "Exploring Different Types of Trust Propagation", *Trust Management*, Vol. 3986/2006 of *Lecture Notes in Computer Science*, pp. 179–192, 2006.

122. Guha, R., R. Kumar, P. Raghavan and A. Tomkins, "Propagation of trust and distrust", *Proceedings of the 13th international conference on World Wide Web*, New York, NY, USA, 17-22 May, ACM, pp. 403 – 412, 2004.

123. Zhang, B., Y. Xiang and Q. Xu, "Semantics Based Information Trust Computation and Propagation Algorithm for Semantic Web", *5th International Conference on Wireless Communications, Networking and Mobile Computing, WiCom '09*, Beijing, China, 24-26 September, IEEE, pp. 1-4, 2009.

124. Quercia, D., S. Hailes and L. Capra, "Lightweight Distributed Trust Propagation", *Proceedings of the 2007 Seventh IEEE International Conference on Data Mining*, Omaha, NE, 28-31 October, IEEE, pp. 282–291, 2007.

125. Kim, H. K., J. K. Kim and Y. U. Ryu, "Personalized Recommendation over a Customer Network for Ubiquitous Shopping", *IEEE Transactions on Service Computing*, Vol. 2, No. 2, pp. 140–151, 2009.

126. Fouss, F., A. Pirotte, J.-M. Renders and M. Saerens, "Random-Walk Computation of Similarities between Nodes of a Graph with Application to Collaborative Recommendation", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 19, No. 3, pp. 355–369, 2007.

127. Shin, D., J. Lee and S. Yeon, J. Lee, "Context-Aware Recommendation by Aggregating User Context", *2009 IEEE Conference on Commerce and Enterprise Computing*, Vienna, Austria, 20-23 July, IEEE Computer Society, pp. 423–430, 2009.

128. Huang, Z., J. Huai, H. Sun, X. Liu and H. Li, "BestRec: A Behavior Similarity Based Approach to Services Recommendation", *2009 Congress on Services - I*, Los Angeles, CA, 6-10 July, IEEE, pp. 46–53, 2009.