

IFIP AICT 300



Elena Ferrari  
Ninghui Li  
Elisa Bertino  
Yuecel Karabulut  
(Eds.)

# Trust Management III

Third IFIP WG 11.11  
International Conference, IFIPTM 2009  
West Lafayette, IN, USA, June 2009  
Proceedings

 Springer



## **IFIP – The International Federation for Information Processing**

IFIP was founded in 1960 under the auspices of UNESCO, following the First World Computer Congress held in Paris the previous year. An umbrella organization for societies working in information processing, IFIP's aim is two-fold: to support information processing within its member countries and to encourage technology transfer to developing nations. As its mission statement clearly states,

*IFIP's mission is to be the leading, truly international, apolitical organization which encourages and assists in the development, exploitation and application of information technology for the benefit of all people.*

IFIP is a non-profitmaking organization, run almost solely by 2500 volunteers. It operates through a number of technical committees, which organize events and publications. IFIP's events range from an international congress to local seminars, but the most important are:

- The IFIP World Computer Congress, held every second year;
- Open conferences;
- Working conferences.

The flagship event is the IFIP World Computer Congress, at which both invited and contributed papers are presented. Contributed papers are rigorously refereed and the rejection rate is high.

As with the Congress, participation in the open conferences is open to all and papers may be invited or submitted. Again, submitted papers are stringently refereed.

The working conferences are structured differently. They are usually run by a working group and attendance is small and by invitation only. Their purpose is to create an atmosphere conducive to innovation and development. Refereeing is less rigorous and papers are subjected to extensive group discussion.

Publications arising from IFIP events vary. The papers presented at the IFIP World Computer Congress and at open conferences are published as conference proceedings, while the results of the working conferences are often published as collections of selected and edited papers.

Any national society whose primary activity is in information may apply to become a full member of IFIP, although full membership is restricted to one society per country. Full members are entitled to vote at the annual General Assembly, National societies preferring a less committed involvement may apply for associate or corresponding membership. Associate members enjoy the same benefits as full members, but without voting rights. Corresponding members are not represented in IFIP bodies. Affiliated membership is open to non-national societies, and individual and honorary membership schemes are also offered.

Elena Ferrari Ninghui Li Elisa Bertino  
Yuecel Karabulut (Eds.)

# Trust Management III

Third IFIP WG 11.11 International Conference, IFIPTM 2009  
West Lafayette, IN, USA, June 15-19, 2009  
Proceedings

Volume Editors

Elena Ferrari

University of Insubria, Department of Computer Science and Communication

Via Mazzini, 5, 21100 Varese, Italy

E-mail: elena.ferrari@uninsubria.it

Ninghui Li

Elisa Bertino

Purdue University, Department of Computer Science

305 N. University Street, West Lafayette, IN 47907-2107, USA

E-mail: {ninghui, bertino}@cs.purdue.edu

Yuecel Karabulut

SAP's Office of the CTO

3412 Hillview Avenue, Palo Alto, CA 94304, USA

E-mail: yuecel.karabulut@sap.com

Library of Congress Control Number: Applied for

CR Subject Classification (1998): D.4.6, K.6.5, C.2, K.4.4, E.3

ISSN 1868-4238

ISBN-10 3-642-02055-0 Springer Berlin Heidelberg New York

ISBN-13 978-3-642-02055-1 Springer Berlin Heidelberg New York

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

springer.com

© International Federation for Information Processing 2009

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper SPIN: 12689507 06/3180 5 4 3 2 1 0

# Preface

This volume contains the proceedings of IFIPTM 2009, the Third IFIP WG 11.11 International Conference on Trust Management, held at Purdue University in West Lafayette, Indiana, USA during June 15–19, 2009.

IFIPTM 2009 provided a truly global platform for the reporting of research, development, policy and practice in the interdependent areas of privacy, security, and trust. Building on the traditions inherited from the highly successful iTrust conference series, the IFIPTM 2007 conference in Moncton, New Brunswick, Canada, and the IFIPTM 2008 conference in Trondheim, Norway, IFIPTM 2009 focused on trust, privacy and security from multidisciplinary perspectives. The conference is an arena for discussion about relevant problems from both research and practice in the areas of academia, business, and government.

IFIPTM 2009 was an open IFIP conference. The program of the conference featured both theoretical research papers and reports of real-world case studies. IFIPTM 2009 received 44 submissions. The Program Committee selected 17 papers for presentation and inclusion in the proceedings. In addition, the program and the proceedings include one invited paper and five demo descriptions. The highlights of IFIPTM 2009 included invited talks and tutorials by academic and governmental experts in the fields of trust management, privacy and security, including Eugene Spafford, Marianne Winslett, and Michael Novak.

Running an international conference requires an immense effort from all parties involved. We would like to thank the Program Committee members and external referees for having provided timely and in-depth reviews of the submitted papers. We would also like to thank the Workshop, Tutorial, Demonstration, Local Arrangements, and Website Chairs, for having provided great help organizing the conference.

In addition, we appreciate the logistics support provided by the Center for Education and Research in Information Security and Assurance (CERIAS) at Purdue University. We are also grateful to the SAP Office of the CTO for their financial support for IFIPTM 2009.

We hope you enjoy the proceedings.

June 2009

Elena Ferrari  
Ninghui Li  
Elisa Bertino  
Yuecel Karabulut



## Christian

Damsgaard Jensen	Technical University of Denmark, Denmark
Anupam Datta	Carnegie Mellon University, USA
Theo Dimitrakos	BT Innovate, UK
Naranker Dulay	Imperial College London, UK
Sandro Etalle	T.U. Eindhoven and University of Twente, The Netherlands
Rino Falcone	Institute of Cognitive Sciences and Technologies of CNR Rome, Italy
Giusella Finocchiaro	University of Bologna, Italy
Ehud Gudes	Ben Gurion University of the Negev, Israel
Peter Herrmann	The Norwegian University of Science and Technology, Norway
Valerie Issarny	INRIA, France
Audun Jøsang	University of Oslo, Norway
James Joshi	University of Pittsburgh, USA
Günter Karjoth	IBM Zurich Research Laboratory, Switzerland
Larry Koved	IBM T.J. Watson Research Center, USA
Mark Kramer	MITRE, USA
Adam J. Lee	University of Pittsburgh, USA
Javier Lopez	University of Malaga, Spain
Stéphane Lo Presti	Brunel University, UK
Stephen Marsh	NRC Canada, Canada
Fabio Martinelli	IIT-CNR, Italy
Fabio Massacci	University of Trento, Italy
Yuko Murayama	Iwate Prefectural University, Japan
Babak Sadighi	Axiomatics AB, Sweden
Kent Seamons	Brigham Young University, USA
Jean-Marc Seigneur	University of Geneva, Switzerland
Mark Ryan	University of Birmingham, UK
Simon Shiu	HP Labs, UK
Jessica Staddon	PARC, USA
Ketil Stølen	SINTEF and University of Oslo, Norway
Vipin Swarup	MITRE, USA
Sotirios Terzis	University of Strathclyde, UK
Bhavani	
Thuraisingham	University of Texas at Dallas, USA
Mahesh Tripunitara	University of Waterloo, Canada
William H.	
Winsborough	University of Texas at San Antonio, USA
Marianne Winslett	University of Illinois at Urbana-Champaign, USA
Danfeng Yao	Rutgers University, New Brunswick, USA



## External Reviewers

Isaac Agudo  
Sruthi Bandhakavi  
Roberto Speicys Cardoso  
Piotr Cofta  
Heidi Dahl  
Tien Tuan Anh Dinh  
Nicola Dragoni  
Carmen Fernandez-Gago  
Nurit Galoz  
Ragib Hasan  
Tormod Havaldsrud

Aliksandr Lazouski  
Pierre De Leusse  
Thomas Martin  
Ruben Rios  
Fredrik Seehusen  
Tamas Visegrady  
Mozhgan Tavakolifard  
Emmanuele Zambon  
Nicola Zannone  
Charles Zhang

## Sponsoring Institutions

SAP Office of the CTO

# Table of Contents

## Social Aspects and Usability

Spiral of Hatred: Social Effects in Buyer-Seller Cross-Comments Left on Internet Auctions . . . . .	1
<i>Radoslaw Nielek, Aleksander Wawer, and Adam Wierzbicki</i>	
Graphical Passwords as Browser Extension: Implementation and Usability Study . . . . .	15
<i>Kemal Bicakci, Mustafa Yuceel, Burak Erdeniz, Hakan Gurbaslar, and Nart Bedin Atalay</i>	

## Trust Reasoning and Processing

Trust-Enhanced Recommender Systems for Efficient On-Line Collaboration . . . . .	30
<i>Georgios Pitsilis</i>	
Towards Understanding the Requirements and Limitations of Reputation-Based Systems . . . . .	47
<i>Mohamed Ahmed and Stephen Hailes</i>	
Elimination of Subjectivity from Trust Recommendation . . . . .	65
<i>Omar Hasan, Lionel Brunie, Jean-Marc Pierson, and Elisa Bertino</i>	

## Data Security

Security in Wiki-Style Authoring Systems . . . . .	81
<i>Christian Damsgaard Jensen</i>	
On Usage Control in Data Grids . . . . .	99
<i>Federico Stagni, Alvaro Arenas, Benjamin Aziz, and Fabio Martinelli</i>	
Detection and Prevention of Insider Threats in Database Driven Web Services . . . . .	117
<i>Tzvi Chumash and Danfeng Yao</i>	

## Enhancements to Subjective Logic

Inferring Trust Based on Similarity with TILLIT . . . . .	133
<i>Mozhgan Tavakolifard, Peter Herrmann, and Svein J. Knapskog</i>	
Analogical Trust Reasoning . . . . .	149
<i>Mozhgan Tavakolifard, Peter Herrmann, and Pinar Öztürk</i>	

## Information Sharing

TIUPAM: A Framework for Trustworthiness-Centric Information Sharing . . . . .	164
<i>Shouhuai Xu, Ravi Sandhu, and Elisa Bertino</i>	
TrustBuilder2: A Reconfigurable Framework for Trust Negotiation . . . . .	176
<i>Adam J. Lee, Marianne Winslett, and Kenneth J. Perano</i>	
A Relational Wrapper for RDF Reification . . . . .	196
<i>Sunitha Ramanujam, Anubha Gupta, Latifur Khan, Steven Seida, and Bhavani Thuraisingham</i>	

## Risk Assessment

Employing Key Indicators to Provide a Dynamic Risk Picture with a Notion of Confidence . . . . .	215
<i>Atle Refsdal and Ketil Stølen</i>	
A Risk Based Approach to Limit the Effects of Covert Channels for Internet Sensor Data Aggregators for Sensor Privacy . . . . .	234
<i>Camilo H. Viecco and L. Jean Camp</i>	

## Simulation of Trust and Reputation Systems

An Experimental Testbed for Evaluation of Trust and Reputation Systems . . . . .	252
<i>Reid Kerr and Robin Cohen</i>	
Evaluating the STORE Reputation System in Multi-Agent Simulations . . . . .	267
<i>Jonas Andrulis, Jochen Haller, Christof Weinhardt, and Yuecel Karabulut</i>	
Comparison of the Beta and the Hidden Markov Models of Trust in Dynamic Environments . . . . .	283
<i>Marie E.G. Moe, Bjarne E. Helvik, and Svein J. Knapskog</i>	

## Demonstration Abstracts

WRS: The Wikipedia Recommender System . . . . .	298
<i>Thomas Lefèvre, Christian Damsgaard Jensen, and Thomas Rune Korsgaard</i>	
Distributed Systems Security Governance, a SOA Based Approach . . . . .	302
<i>Pierre de Leusse and David Brossard</i>	

Security and Trust Management for Virtual Organisations: GridTrust Approach . . . . .	306
<i>Syed Naqvi and Paolo Mori</i>	
Common Capabilities for Trust and Security in Service Oriented Infrastructures . . . . .	310
<i>David Brossard and Maurizio Colombo</i>	
A Virtual Hosting Environment for Distributed Online Gaming . . . . .	314
<i>David Brossard and Juan Luis Prieto Martinez</i>	
<b>Author Index . . . . .</b>	<b>319</b>

# Spiral of Hatred: Social Effects in Buyer-Seller Cross-Comments Left on Internet Auctions\*

Radoslaw Nielek<sup>1</sup>, Aleksander Wawer<sup>2</sup>, and Adam Wierzbicki<sup>3,\*\*</sup>

<sup>1</sup> Polish Japanese Institute of Information Technology,  
ul. Koszykowa 86, 02-008 Warszawa, Poland  
nielek@pjwtk.edu.pl

<sup>2</sup> Institute of Computer Science Polish Academy of Sciences,  
ul. J.K. Ordon 21, 01-237 Warszawa, Poland  
axw@ipipan.waw.pl

<sup>3</sup> Polish Japanese Institute of Information Technology,  
ul. Koszykowa 86, 02-008 Warszawa, Poland  
adamw@pjwtk.edu.pl

**Abstract.** An auction platform is a dynamic environment where a rich variety of social effects can be observed. Most of those effects remain unnoticed or even hidden to ordinary users. The in-depth studies of such effects should allow us to identify and understand the key factors influencing users' behaviour. The material collected from the biggest Polish auction house has been analyzed. NLP algorithms were applied to extract sentiment-related content from collected comments. Emotional distance between negative, neutral and positive comments has been calculated. The obtained results confirm the existence of the spiral-of-hatred effect but also indicate that much more complex patterns of mutual relations between sellers and buyers exist. The last section contains a several suggestions which can prove useful to improve trustworthiness of users' reports and security of an auction platform in general.

## 1 Introduction

Transaction volumes and numbers of users in e-commerce systems have been booming over the past few years and there is no sign of a slowdown in the foreseeable future. Every new account in an auction house or within web 2.0 services creates new challenges for privacy and security. Auction houses seem to be the most demanding environment for trust management systems due to direct relationship between reputation and users' income [3][4]. Every unpunished and undetected fraud undermines the honest agents' motivation to play fair. Thus, many researchers are working to create new reputation algorithms. Nevertheless, reputation management systems embedded in the most popular websites remain practically unchanged over years and are based on very simple quantitative evaluations and qualitative comments.

---

\* This research has been supported by the Polish Ministry of Science grant N N516 4307 33.

\*\* This author has been supported by the grant of the Polish Ministry of Science 69/N-SINGAPUR/2007/0.

Thus far, most researches have been focused on improving algorithms using qualitative feedback and therefore there is a relatively narrow selection of papers devoted to mining comments (security perspective [2], trustworthiness of reviews [17]) and developing algorithms for trust management systems which explicitly consider descriptive opinions [15]. This is so partly because natural language processing module, which is the cornerstone of such an algorithm, requires building it almost from scratch for every single language (the reusable part is insignificant). It means that the results obtained for languages other than English are hardly comparable and difficult to validate for a larger scientific community.

Nevertheless, it makes good sense to devote resources to the discovery of patterns in descriptive opinions expressed in languages other than English since most Internet transactions are done in language environment native to participants and as local web auction markets grow very fast, this situation will probably continue into the future. Many observations reported in this paper are likely to apply to other cultures too, irrespective of the language in which the comments are written. Primarily, users' behavioural patterns refer to more general psychological (e.g. spiral of hatred - response is stronger than impulse [14]) and sociological effects which can be even stronger than the cultural fingerprint. A comparative study on Taobao (Chinese version of an online auction marketplace) and eBay has partly confirmed this assumption [10].

In reference to the above, this paper is devoted to an analysis of users' behaviour during the after-transaction evaluation process, in particular taking into account pairs of comments on the same transactions delivered by both sellers and buyers (cross-comments) in the auction house. Two approaches have been used to identify and validate different hypotheses. In section 2, feedback mechanisms existing in e-commerce systems are described. Section 3 is devoted to quantitative and statistical examination of the collected data and focuses on the effects related to comment type and order in which they arrive. The results obtained by natural language processing algorithms in the context of the hypothesis for validating the spiral of hatred effect are presented in the fourth section. The fifth section features discussion of the results and a new heuristic model to solve some of the identified problems. The last section presents the conclusions and possible trends in the future research.

## 2 Quantitative and Qualitative Comments

The most commonly used reputation systems embedded in online auction website allow us to evaluate transaction results not only by selecting a predefined category from a list but also by leaving shorter or longer comments. The quantitative measurement in use by eBay and Allegro (the biggest Polish auction house) is based on a very simple structure. When a given transaction is completed, every eBay/Allegro user can evaluate his or her partner by choosing either a positive or neutral, or negative mark. The evaluation mark is visible after being submitted. On eBay it is also possible to evaluate separately the quality of a delivered product, communication, shopping time as well as shipping and handling charges. All those additional evaluation are anonymous. The sums of positive, negative and neutral marks are presented separately. Because feedback is not obligatory, not every transaction is followed by its evaluation. As shown in [1] no information is usually indicative of bad experience during the transaction.

Predominantly, only positive comments appear. For more than 1.7 million comments in the collected database there were only ca. 9000 negative and ca. 5000 neutral comments which means that either the fraud level is very low or (it seems more likely) there is a mechanism, which discourages people from making negative comments. Certainly, the threat of legal action [8] constitutes one source of fear, another one is probably related to the possibility of being punished with negative reciprocal evaluation. Yet, another effect identified by researchers [7] is that users award a positive quantitative evaluation mark but describe all negative aspects of a transaction in words.

The relative stable framework in the auction houses provides a good opportunity to detect even quite complicated users' behavioural patterns. Abilities of users to learn from previous experiences and to modify their strategies appear to be non-trivial attractors within the space of possible behavioural patterns. A good example of self-adaptation in the complex system which has emerged in online auction websites is that users pay much more attention to negative comments when they calculate transaction risk [11].

Typically, users can intentionally express their opinions only by making comments which are composed of a selected label (quantitative) and a description (qualitative). Nevertheless, a lot of additional information can be found in the data collected in the online auction website, for example response times on positive and negative evaluations, order of buyer-seller evaluation, length of comments or context and reference points (average rating for specific subsets). Identifying measurable effects in buyer-seller interaction can help improve the existing trust management algorithms and create a foundation for designing new ones.

## 3 Experiments

### 3.1 Dataset

The database analysed in this article was provided by the biggest Polish auction house (over 70% of market share). At the beginning of the fourth quarter in 2006, 10,000 sellers and 10,000 buyers have been randomly selected; their profiles and received comments have been stored (description and evaluation). During the next 6 months all transactions conducted by the selected users were monitored and recorded. For every partner who appeared in transaction and was not in the primary database, all historical information about the received feedback has been collected, but with respect to new auctions only the originally selected users have been monitored. In the first quarter in 2007 the database contained more than 200,000 transactions and over 1.7 million comments.

#### 3.1.1 Formal Definition

Symbols used in the following sections are defined below:

- $U$  — set of all users,
- $T$  — set of all transaction,
- $t_m$  — m-th transaction,
- $u_i$  — i-th user,
- $c_{t_m}^{u_i}$  — comment left by the i-th user after m-th transaction,

- $\tau(c_{t_m}^{u_i})$  — sentiment measured by sentipejd for the comment  $c$ ,
- $\rho(c_{t_m}^{u_i})$  — label for the comment  $c$  given by the  $i$ th user,
- $r(t_m, u_i)$  — the role of the  $i$ -th user in the  $m$ -th transaction (either buyer or seller),
- $\varphi(c_{t_m}^{u_i})$  — timestamp for the comment  $c$ ,
- $\omega_m = (c_{t_m}^{u_i}, c_{t_m}^{u_j})$  — an ordered pair of comments for the  $m$ -th transaction,
- $\delta(\omega_m)$  — time between two comments,

### 3.1.2 Amount, Type, Time and Order in Cross-Comments

The objective of this paper is to identify the effects which appear during bi-directional evaluation, therefore the main focus was an analysis of the ordered pairs of comments, defined in the previous section as  $\omega_{t_m}$ . For over 1.7 million comments slightly more than 800 thousand pairs were found (in ca. 9% of cases only one party of a given transaction left a comment – either buyer or seller) Only 5056 of pairs contain at least one non-positive evaluation.

Over 90% of answers for comments are made within 14 days after the first evaluation. Shape of curves on the Fig.1 is similar for all considered cases but there is a notable bias in the starting point. In general, sellers are more responsive - for negative and neutral comments over 20% of sellers and only 7% of buyers feedbacks were written in less than one hour after receiving an evaluation from the partner (for positive comments the numbers are 7% and 3% respectively). On average, buyers seem to visit the auction website less often, so their reaction is slower. Comments, regardless of their contents, are emailed to the evaluated user, thus there is no other variable, except for the type of comment, that may explain the variation in reaction times. Very short response times for negative and neutral comments (when compared with positive

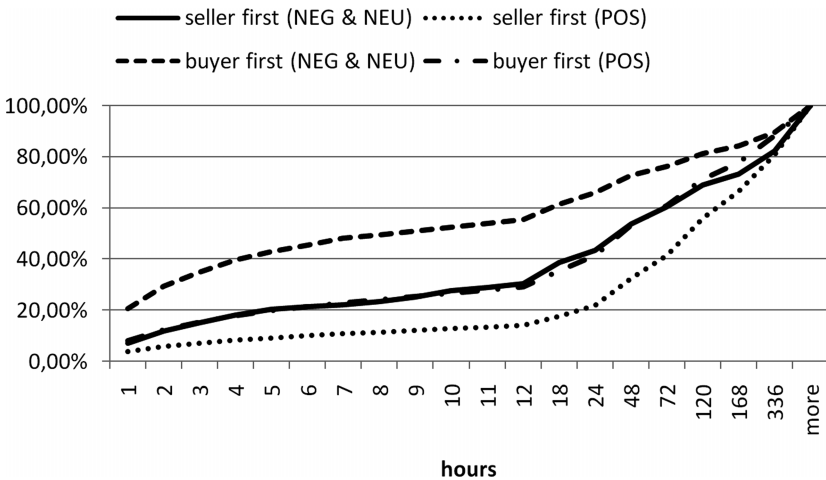


Fig. 1. Time-span between comment and answer for different category of evaluation (cumulative histogram)



**Table 1.** Average length of comments (in characters)

	Buyer	Seller
POS	102.49	73.18
NEU	149.02	154.27
NEG	183.77	178.78

feedbacks) can be explained by the will to punish, as fast as possible, the author of the negative  $\square$  evaluation.

Average length of comments presented in Table 1. varies between both transaction roles and different feedback types. As a rule, longer text creates an opportunity to enumerate more facts and express a broader variety of emotions, but also emphasizes the importance of the particular comment for a given user - she or he has been ready to devote more time to leave feedback. For a positive experience, which is a typically expected result of the transaction, the comments are relatively short - 100 characters in the case of sellers and 73 of buyers (the difference is statistically significant). Manual inspection of a both comment types indicates that the cause of this difference results from the habit of adding an advertisement at the end of comments made by sellers (e.g. *"Hope to see you again. ALFRA\_PL"*).

Dissatisfying transaction outcome is positively correlated with length of evaluations. More characters are needed to describe and probably justify dissatisfaction and the negative feedback. The difference between buyer and seller observed for positive comments disappears for both negative and neutral evaluations (small differences observed in table 1. in the second and third row are statistically insignificant).

The unwritten rule in online auction websites is that buyers make comments first. For the pairs of comments containing only positive evaluation in 8.2% cases this rule was broken. If one of the comments is negative or neutral, the number of cases contradicting the unwritten law rises dramatically to over 18%. There are many reasons why sellers decide to make a comment first. Some of the sellers probably participate in too many transactions to follow which one is already finished and commented and which not. Strong evidence of such behaviour can be seen in Table 2 in the very last column - more than 8% of sellers answered to negative evaluation with a positive one. More detailed manual analysis of these pairs indicates that some of the answers contain explanations of the reason for unsatisfactory quality of service (e.g. limited access to the Internet or problems with logistic) but most is given disregarding the previous

**Table 2.** Combination of comment-answer pairs (buyer first)

	POS/x	NEU/x	NEG/x
x/POS	—	937 (22.60%)	339 (8.17%)
x/NEU	0 (0%)	686 (16.55%)	41 (0.99%)
x/NEG	0 (0%)	408 (9.84%)	1734 (41.83%)

<sup>1</sup> As it is shown in the next sections, neutral comments are very similar to negative comments.

negative comment. Yet another hypothesis is that the seller is forced by an external event to send the feedback- he or she needs to pay commission for the auction website within a limited period of time after finishing the transaction regardless of its outcome.

If the buyer is satisfied and expresses this satisfaction with a positive comment first, the answer from the seller will always be positive. All the collected pairs confirm this rule without exceptions. It could be only partially explained by the previous observation. First, a vast domination of positive comments makes a pair  $\omega_m = (c_{t_m}^{u_i}, c_{t_m}^{u_j})$  such as  $\rho(c_{t_m}^{u_i}) = pos \wedge \rho(c_{t_m}^{u_j}) \neq pos$  statistically very improbable. Yet, the distribution of such comments was asymmetrical between both transaction roles. On one hand, of over such 550 cases exist for  $r(t_m, u_i) = seller$ , on the other, no such comment pair was found for  $r(t_m, u_i) = buyer$ . Secondly, the results[11] show that even substantial amount of negative feedback does not affect the ability of buyers to participate in transactions. Therefore, a positive opinion about the seller is always rewarded with a reciprocal positive feedback. Thirdly, although there is no explicitly defined procedure to change already submitted feedback, it is essentially possible after some reasonable efforts (e.g. sending an email to the webmaster). So, a seller can refrain from making a non-positive evaluation only because of an aversion to initiating a "war", even though not everything went correct during the transaction.

**Table 3.** Combination of comment-answer pairs (seller first)

	POS/x	NEU/x	NEG/x
x/POS	—	0 (0%)	0 (0%)
x/NEU	284 (31.17%)	19 (2.08%)	19 (2.08%)
x/NEG	242 (26.56%)	8 (0.87%)	339 (37.21%)

Ordered pairs of comments  $\omega_m = (c_{t_m}^{u_i}, c_{t_m}^{u_j})$  such as  $r(t_m, u_i) = neu \wedge r(t_m, u_j) = neg$  (the first evaluation is neutral and the second negative) appear eight times more frequently (416) than pairs where  $r(t_m, u_i) = neg \wedge r(t_m, u_j) = neu$  (the first evaluation is negative and the second neutral) (60). This enormous disproportion cannot be explained by the course of the transaction because there is no evidence to claim that a negatively affected party will comment second. A more credible explanation is that neutrally evaluated agents use negative evaluations as a punishment and try to do it as severely as possible.

## 4 Mining the Meaning of Comments

### 4.1 Automatic Sentiment Extraction

For the sentiment analysis task we used a modified version of Sentipejd [20] - a hybrid of lexeme category analysis with a shallow parsing engine. At the basic level, Sentipejd checks for presence of a specific category of lexemes. Such an abstraction originates in content analysis systems, most notably the classic General Inquirer [21]. Lexical categories used in this work include two sets of words (dictionaries): 1580 positive and

1870 negative ones, created by Zetema<sup>2</sup>. Because comment texts are typed in a careless manner, very often completely without diacrits, lexeme recognition was extended with a diacrit guesser. Recognized sentiment lexemes, along with morphosyntactic tags, are analyzed with Spejd - a tool for simultaneous morphosyntactic disambiguation and shallow parsing [19], with a number of rules crafted to recognize multiword opinion patterns and apply sentiment modifying operations.

The Spejd formalism is a cascade of regular grammars. Unlike in the case of other shallow parsing formalisms, the rules of the grammar allow for explicit morphosyntactic disambiguation, independently or in connection with structure building statements, which facilitates the task of the shallow parsing of ambiguous and/or erroneous input.

For the purpose of sentiment analysis we extended the default Spejd's morphosyntactic tagset with a sentiment category expressing properties of positive or negative sentiment. This hybrid approach has been called Sentipejd [20].

Sentiment rules, discussed more extensively in [20], included (but were not limited to) the following:

- Affirmation - an expression of positive sentiment, usually an adverb confirming the sentiment of a positive word and should be treated as strong indications of sentiment (eg. 'I strongly recommend')
- Negation - as simple as the difference between "polecam" ('I recommend') and "nie polecam" ('I do not recommend'). The example generic rule captures also statements including the optional verb 'to be' ([base by]), like "nie jest dobry" ('isn't good').
- Nullification - expressing lack of a certain quality or property (usually of negative sentiment), for example "nie mam zastrzezen" ('I have no objections').
- Limitation - a limiting expression tells us that an expression of positive and negative sentiment has only a very limited extend, therefore hinting that the general sentiment of the review is the opposite of the expression. Example: "jedyny problem" ('the only problem').
- Negative modification - an adjective of negative sentiment preceding a positive noun, for example "koszmarne jakosc" ('nightmarish quality').

Sentipejd returns either vectors of two integers (emoi=[pos, neg]) which express separately strengths of positive and negative emotions (it's not a simple sum of all emotional phrases) or the single, composite value  $-\tau(c_{tm}^{t_i})$ . Every comment present in the collected dataset has been analyzed separately and the result has been stored as a vector in a database together with a category of the comment and the comment itself.

## 4.2 Reclassification Precision and the Emotional Distance

Although a similar natural language processing module has been already applied by authors to a broad variety of subjects (e.g. dynamic of public opinion[5]) the very first question which arises is: can a NLP system extract and evaluate emotions from usually very short and not always correctly (grammatical mistakes and typos) written comments? To answer this question, which is crucial for further deliberations, a standard data mining approach was used.

<sup>2</sup> www.zetema.pl

Four separated, balanced subsets of comments were created:

- Set I (POS; NEG) - contains 2590 comments whereof 1295 are negative and 1295 positive,
- Set II (NEG; NEU) - contains 1454 comments whereof 727 are negative and 727 neutral,
- Set III (NEU; POS) - contains 1454 comments whereof 727 are neutral and 727 positive,
- Set IV (POS; NEU; NEG) - contains 2181 comments whereof 727 are positive, 727 negative and 727 neutral,

Every set of comments has been partitioned on testing and training set (30% and 70% cases respectively). For every set of comments three different classification approaches were used: neural network, support vector machine and decision trees (CHAID algorithm). As a target variable the label given by comment's author (negative, neutral or positive) was selected and the emo vector as input variables.

**Table 4.** Classification accuracy for different algorithms and testing subsets (average of four runs)

	Neural Network	Support Machine	Vector Decision (CHAID)	Trees
POS and NEG (two classes; set I)	90,86%	90,36%	89,37%	
POS, NEU and NEG (three classes; set IV)	61,58%	61,66%	61,79%	
NEG and NEU (two classes; set II)	65,16%	65,80%	64,11%	
NEU and POS (two classes; set III)	71,15%	69,15%	70,24%	

The obtained results are presented in table 4. The first experiment was conducted to check if an evaluation based on the emotions expressed in comments and measured by the Sentipejd allows to predict the polarity of an label given by a human. At the beginning, the simplest subset was tested (only two classes - positive and negative - which should be relatively easier to separate). For the first set neural network approach was the most efficient. Over 90% classification accuracy indicates that the Sentipejd deals quite well with extracting emotions from texts (even not 100% correctly written) and that the significant difference in emotional content between positive and the negative labelled comments can be confirmed and measured.

Similar results for the neural network, support vector machines and decision trees (90.86%, 90.36% and 89.37% respectively) suggest that the reason for wrong classification goes beyond the classification algorithms. Only slightly better results for the same algorithms but validated on training sets instead of test sets seem to confirm that as well. A closer look at the misclassified cases shows that they belong into three (not always distinct) groups:

- written in a very specific slang, many misspellings, grammatical and orthographical errors, a lot of emoticons,
- well written but based on ironic, quizzical description of the past transaction,
- marked by user as positive but containing a negative evaluation,

The existence of the third group seems to confirm the results presented by Botsch and Luckner [7]. Some users, instead of leaving a negative mark, prefer to describe all the experienced problems in words. Because of their incoherency, those cases cannot be correctly classified using the adopted approach and they should be removed from the database. A detailed estimation of the scale of this effect requires manual processing of every comment which is not feasible because of the database size (1.7 million comments) and extends beyond the scope of this paper, although a rough estimation indicates that the effect of incoherent feedbacks is lower than 0.1% of all positive comments.

The biggest fraction of wrongly classified comments belongs to the first group. Many users, not only in e-Commerce systems but also on online forums, use a lot of abbreviations, emoticons, colloquial words and even intentionally misspelled words. Frequently, using intentionally transformed words is a sign of being a member of a specific social group. It helps users to identify the newcomers in an environment where cheap pseudonyms are present (a detailed study of the effects introduced by using cheap pseudonyms can be found here[16]). Some problems can be resolved (e.g. using a spell-checker to correct orthographical mistakes or creating a dedicated dictionary containing slang and colloquial words) but in principle intentional modifications of meaning or detecting irony will always be a challenge for computational linguistics.

The results for set IV are presented in the second row in table 4. Introduction of the third class made the task much more difficult. The results over 60% are still almost 30% better than in the baseline of random choice but significantly lower than for two classes. Thus, to check which comments cause problems for the classification algorithms, two more experiments have been conducted. Firstly, the separability for neutral and negative comments has been tested. The third row in table 4 contains the results for the set II which includes only negative and neutral comments. The classification precision slightly over 65% indicates that the emotional distance between neutrally and negatively tagged feedback is relatively small. Secondly, the same approach has been used to measure the emotional distance between neutral and positive comments. The results for all classification methods except support vector machines are at least 6% better and indicate that neutral comments are emotionally closer to negative.

To confirm the hypothesis stated in the previous paragraph a new testing set has been created. All the collected comments were split into two classes: one containing only positive labelled comments and one with negative and neutral feedback. Based on the emo vector (defined at the beginning of this section) and using the classification algorithms (support vector machines, neural network and decision trees) an attempt to rediscover the new classification has been done. The obtained results are slightly less precise than for the set I (positive and negative comments only; without neutrals) but the difference is about 3%. Thus, in most applications negative and neutral comments can be interpreted in the same way - as an expression of dissatisfaction. The label should not be treated as a scale of the experiences) because there is very little data to confirm the hypothesis that neutral feedback is less effective than negative.

### 4.3 Spiral of Hatred

The spiral of hatred is a well-known phenomenon present in a wide area of scientific fields (eg. Wydra identifies it as an core component of the war conflicts[22]) manifested as an endless action-reaction response, where successive iterations are subject to more negative emotion. Typically, in practical terms this effect can be observed on online forums where an initial misunderstanding causes a lasting exchange of messages containing many abusive words. Because reputation influences profitability of the seller[11] and every negative comment undermines this reputation, thus the reaction of a seller after receiving negative feedback can be more emotional. In fact, as a consequence of unbalanced levels of positive and negative comments, a very interesting heuristic has emerged. For experienced users, a single negative comment plays much more important role in the estimation of transaction risk that even many positive comments.

Because comments are visible after they are left, a natural place to express (and observe), the spiral-of-hatred effect is the reciprocal feedback given by the second party after transaction is completed. Thus, the database described in section 3.1. has been used to verify the spiral-of-hatred effect, which – referring to the formalism defined in section 3.2 - can be expressed as:

$$\forall \rho(c_{t_m}^{u_i}), \rho(c_{t_m}^{u_j}) \in \{NEG, NEU\} : \varphi(c_{t_m}^{u_i}) < \varphi(c_{t_m}^{u_j}) \rightarrow \tau(c_{t_m}^{u_i}) > \tau(c_{t_m}^{u_j}) \quad (1)$$

It is reasonable to assume (because of the sociological nature of the analyzed effect) that the above definition will not apply universally and to every single case. Therefore, in the first stage a weaker assumption was tested – the average of negative emotion for the second comment is higher than for the first:

$$\forall w, u \in U; s, t \in T : (\rho(c_s^u), \rho(c_t^w) \neq POS : \varphi(c_t^w) < \varphi(c_s^u)) \rightarrow \sum_{w,t} \tau(c_t^w) < \sum_{u,s} \tau(c_s^u) \quad (2)$$

The results are equivocal. First, the average value of  $\tau$  for the comments given first is  $-0.63$ . The same value for the answers is higher and amounts  $-0.72$ . The difference is statistically significant at the level 0.07 which is a little bit above a typical 0.05 but it seems to make a spiral of hatred hypotheses at least very probable. Second, the standard deviations for both sets are almost equal – 2.01 – and it indicates that the distribution of emotion intensity between both the earlier and latter comment groups is similar but shifted. On the other hand, dividing the set analyzed in the previous paragraph into buyer and seller roles of the agent, makes results more complicated. More detailed results are presented in table 5.

**Table 5.** Average sentiment for buyers and sellers

	first	second
buyer	-0.55	-0.69
seller	-1.28	-0.96

In general, sellers are more emotional and more expressive than buyers ( $\tau = -0.76$  as compared to  $\tau = -0.60$  for buyers) and this pattern concerns both specific cases analyzed in table 3. There are at least three hypotheses which can explain this difference. Firstly, sellers write more correctly so the Sentipejd has an easier job extracting emotions from comments. Secondly, the cost of receiving negative evaluations for sellers is much higher (pseudonyms are more expensive and lower reputation affects profitability) and therefore boosts their reaction. Thirdly, sellers are simply more experienced and know how to make comments in a more negative way. As the standard deviation for sellers is only slightly higher than for buyers (should be significantly higher, if the source of difference in the emotional strength is related to misspellings and errors in comments), the second and the third hypotheses are the more probable ones.

The classical definition of the spiral-of-hatred effect formalized in eq. 1 and 2 is satisfied (and statistically significant) only for typical cases where buyers leave comments first. As expected, the average answer given by a seller is more negative. However, the same assumption is not true for the uncommon situation where sellers comment first. In that case the average negative sentiment in ordered pairs  $\omega_m = (c_{t_m}^{u_i}, c_{t_m}^{u_j})$  such as  $r(c_{t_m}^{u_i}) = \text{seller}$  is  $-1.12$  and is much higher ( $-0.96$ ) than for  $r(c_{t_m}^{u_j}) = \text{buyer}$ . The increase in negative emotions, compare to situation where buyers comment first, is observed symmetrically for both participants (buyers and sellers). Even though buyers answer very aggressively, at the end the emotional war is always won by sellers. They have stronger motivation because the reputation affects their profitability and are more experienced due to the extensive usage of the auction website.

More studies are needed to determine how the communication beyond auction platforms' cross-comments mechanism (e.g. via e-mail) influences emotional attitudes. However, on the very basic level the spiral-of-hatred effect can be identified in the collected data despite complex interactions of many social processes.

## 5 Discussion

Originally, the auction houses have been developed as goods exchange platforms where everyone could be either a seller or a buyer and where such roles are volatile and adopted only for one transaction. Nowadays, the auction platforms remind more of a shopping mall rather than a medieval bazaar and almost all members have clearly defined typical roles of either sellers or buyers. Therefore, it is necessary to revise the previous paradigm which used to determine the development of the reputation management systems. Instead of two more or less equal transaction parties, there is an explicit distinction: on one hand, sellers become more experienced due to the extensive usage of the auction system, on the other hand buyers' profitability is less sensitive to negative feedback.

The modification of the reputation system should take into consideration these facts. One of the possible ways to take them into account is for example to limit the possibility of leaving an evaluation by making it available only for buyers. One-sided comments make sellers defenceless, but elimination of negative reciprocal feedback will increase the likelihood that buyers comment more honestly. As an undesirable side effect of such a situation, blacklists of dishonest buyers can be created and maintained outside auction

platforms, which can in turn be used as a tool for sometimes unjustified discrimination. More side effects should also be expected.

Another way to eradicate the spiral-of-hatred effect, which requires merely a minor modification in the existing reputation management systems, is to hold back the publication of an evaluation until an answer is sent. It should permit the elimination of the threat of revenge and thus make all comments more honest and less biased by the previous evaluation (more an answer based only on transaction experiences than an evaluation of the other participant performance). The problem that users will intentionally block publication of the negative comments can be solved by introducing a moderator who will be responsible for making an opinion visible (upon a request of one of the transaction parties) even if the answer does not appear. Even fewer changes are required to reduce the identified effect through establishing the minimum time-span that has to pass between comment and answer. Answers given right after a negative comment is received are more emotional and usually less informative.

Natural language processing tools are the best solution to investigate problems referred to in the descriptive part of submitted comments. Automatic sentiment extraction helps identify emotional wars immediately after they appear and either inform the administrators or even take appropriate steps automatically.. Analysis of every pair of comments can be complemented by the knowledge about typical behaviour of users taking part in transaction on the basis of their previous evaluations. Moreover, an efficient NLP algorithm can detect many discrimination strategies such as using a multitude of fake pseudonyms or atypical positive evaluations.

## 6 Conclusion

The broad variety of effects identified and described in this paper is only a fraction of all effects in auction websites. Jointly, with the stoning, slipping, self-selection[9], cheap pseudonyms[16], asymmetrical impact of positive and negative comments[11], price-reputation correlation[12], the importance of missing feedback[1], the presented results provide environment for invention, development and implementation of new techniques and tools with a goal to further increase satisfaction and usability of an auction website. Proposed changes can impact not only users' satisfaction but also profitability of the auction website.

The complex relationships between different users' behavioural patterns and hardly predictable side-effects discourage the managers responsible for maintaining and developing e-commerce systems from modifying the existing, proved solutions. They tend to use simple financial instruments like insurances or escrows to increase the level of security. Thus, the attempts to popularise the results collected by researchers over the last few years should be focused on the development of dedicated external tools to support users using those systems rather than on the modification of existing e-commerce systems.

Future research should be oriented toward sensitivity analysis of identified effects and influence of cultural circles and individual characteristics on the dynamics and existence of particular effects. Also, forecasting of social acceptance and social effects of the planned changes in an auction house is a challenging task[18]. Successful modeling



and forecasting social responses (i.e. emergent attractors, stability points, non-linear dynamics) will be crucial to implement changes in Web 2.0 services.

## References

1. Morzy, M., Wierzbicki, A.: The Sound of Silence: Mining Implicit Feedbacks to Compute Reputation. In: Spirakis, P.G., Mavronicolas, M., Kontogiannis, S.C. (eds.) WINE 2006. LNCS, vol. 4286, pp. 365–376. Springer, Heidelberg (2006)
2. Gregg, D.G., Scott, J.: A Typology of Complaints About Ebay Sellers. *Communications of The ACM* 51(4) (April 2008)
3. Lucking- Reiley, D., Bryan, D., Prasa, N., Reeves, D.: Pennies from eBay: The Determinants of Price in Online Auctions (1999), <http://eller.arizona.edu/~reiley/papers/PenniesFromEBay.pdf>
4. Bajari, P., Hortacsu, A.: Winner's Curse, Reserve Prices and Endogenous entry: Empirical Insights from eBay Auctions. Stanford Institute for Economic Policy Research. SIEPR. Policy paper No. 99-23 (1999)
5. Wawer, A., Nielek, R.: Application of Automated Sentiment Extraction from Text to Modeling of Public Opinion Dynamics. *Journal of Environmental Studies* 17(3B), 508–513 (2008) (in Polish)
6. Resnick, P., Zeckhauser, R.: Trust Among Strangers in Internet Transactions: Empirical Analysis of eBay's Reputation System. *The Economics of the Internet and E-Commerce* 11, S127–S157 (2002)
7. Botsch, J., Luckner, S.: Empirische Analyse von Bewertungskommentaren des Reputationssystems von eBay. Multikonferenz Wirtschaftsinformatik (2008) [DBLP:conf/mkwi/BotschL08]
8. Nam sued for libel over comments on eBay, <http://www.telegraph.co.uk/news/uknews/3247683/Man-sued-for-libel-over-comments-on-eBay.html>
9. Khopkar, T., Li, X., Resnick, P.: Self-selection, slipping, salvaging, slacking, and stoning: the impacts of negative feedback at eBay. In: ACM Conference on Electronic Commerce 2005, pp. 223–231 (2005) (DBLP:conf/sigecom/KhopkarLR05)
10. Lin, Z., Li, J.: The online auction market in China: a comparative study between Taobao and eBay. In: ICEC 2005, pp. 123–129 (2005)
11. Standifird, S.S.: Reputation and E-commerce: eBay Auctions and the Asymmetrical Impact of Positive and Negative Ratings. *Journal of Management* 27(3), 279–295 (2001)
12. Lee, Z., Im, I., Lee, S.J.: The Effect of Negative Buyer Feedback on Prices in Internet Auction Markets. In: Orlikowski, W., et al. (eds.) *The Proceedings of the 21st International Conference on Information Systems*, pp. 286–287 (2000)
13. Ozacka, M., Lim, Y.: A study of reviews and ratings on the internet. In: CHI 2006 Extended Abstracts on Human Factors in Computing Systems, Montréal, Québec, Canada, April 22 - 27, 2006, pp. 1181–1186. ACM, New York (2006), <http://doi.acm.org/10.1145/1125451.1125673>
14. Scheff, T., Fearon Jr, D.: Social and Emotional Components in Self-Esteem. *Journal of the Theory of Social Behavior* 34, 73–90 (2004)
15. Ganesan, K.A., Sundaresan, N., Deo, H.: Mining tag clouds and emoticons behind community feedback. In: *Proceeding of the 17th international Conference on World Wide Web, WWW 2008, Beijing, China, April 21 - 25, 2008*. ACM, New York (2008)
16. Friedman, E.J., Resnick, P.: The Social Cost of Cheap Pseudonyms. *Journal of Economics and Management Strategy* 10, 173–199 (2001)

17. Talwar, A., Jurca, R., Faltings, B.: Understanding user behavior in online feedback reporting. In: Proceedings of the 8th ACM Conference on Electronic Commerce, EC 2007, pp. 134–142. ACM, New York (2007)
18. Kleinberg, J.: The convergence of social and technological networks. *Commun. ACM* 51(11), 66–72 (2008)
19. Buczynski, A., Przepiorkowski, A.: Demo: An Open Source Tool for Partial Parsing and Morphosyntactic Disambiguation. In: Proceedings of LREC 2008 (2008)
20. Buczynski, A., Wawer, A.: Shallow parsing in sentiment analysis of product reviews. In: Proceedings of the Partial Parsing workshop at LREC 2008, pp. 14–18 (2008)
21. Philip, J., et al.: *The General Inquirer: A Computer Approach to Content Analysis*. MIT Press, Cambridge (1966)
22. Wydra, H.: The Recurrence of Violence. *Sociology Compass* 2(1), 183–194 (2008)

# Graphical Passwords as Browser Extension: Implementation and Usability Study\*

Kemal Bicakci<sup>1</sup>, Mustafa Yuceel<sup>1</sup>, Burak Erdeniz<sup>2</sup>, Hakan Gurbaslar<sup>2</sup>,  
and Nart Bedin Atalay<sup>3</sup>

<sup>1</sup> TOBB University of Economics and Technology, Ankara, Turkey  
{bicakci,myuceel}@etu.edu.tr

<sup>2</sup> Middle East Technical University, Ankara, Turkey  
{burakerdeniz,gurbaslar}@gmail.com

<sup>3</sup> Selcuk University, Konya, Turkey  
nartbedin@gmail.com

**Abstract.** Today, most Internet applications still establish user authentication with traditional text based passwords. Designing a secure as well as a user-friendly password-based method has been on the agenda of security researchers for a long time. On one hand, there are password manager programs which facilitate generating site-specific strong passwords from a single user password to eliminate the memory burden due to multiple passwords. On the other hand, there are studies exploring the viability of graphical passwords as a more secure and user-friendly alternative. In this paper, we present GPEX, a password manager program implemented as a web browser plug-in to enable using graphical passwords to secure Internet applications without any need to change their authentication interface. Experimental results show that GPEX has security and usability advantages over other password manager plug-ins. specifically; we find that with the visual interface of GPEX, users have a more complete and accurate mental model of the system and incorrect login attempts causing security exposures can easily be avoided.

**Keywords:** Graphical password, password manager, usable security, authentication.

## 1 Introduction

User authentication is a central component of almost all security applications. The weaknesses of using text based passwords for authentication are well known and there is a significant body of recent research exploring the feasibility of graphical approaches to provide a more secure and usable alternative. Based on the studies showing that human brain is better at recalling images than text [1], graphical passwords are intended to solve memory burden and small password space problem of classical passwords [2].

---

\* This research is supported by TUBİTAK (The Scientific and Technological Research Council of Turkey) under project number 107E227.

Another solution to generate strong passwords is password managers. These manager programs can be implemented as plug-ins to web browsers and they translate easy to remember and low-entropy passwords into stronger passwords which are more immune to dictionary attacks [3-5]. Password managers also provide a solution to “password reuse” problem and are capable of generating site-specific passwords from the same password entry to protect victims against phishing attacks [3-5].

While password managers have attractive security properties, independent usability studies found that they suffer from major usability problems [5]. In particular, one critical finding is that users have inaccurate or incomplete mental models of the password manager software which potentially causes security exposures [5].

In this paper, we join the forces of graphical passwords and password manager programs in order to obtain a more usable and secure authentication system. For this purpose, we use click-based graphical passwords as the new interface of the password manager browser plug-in. More precisely, in the proposed system user clicks several times on an image as a password and the browser extension converts the graphical password into a site-specific text-based password. We think that this system provides a clearer mental model since the translation of graphical password to a text-based one and inserting it to the password field is an easier process to understand. The extension implemented is user-friendly and provides a more secure user experience. For instance, in our system it is obvious when the plug-in has been activated and is awaiting input, thus the solution alleviates the problems associated with incorrectly assumed state of the system. We also argue that our work will ease quick adoption of graphical passwords since it does not require any change on the server side.

Implementing a browser extension for click-based graphical passwords was not as straightforward as we initially thought. We discovered that a design allowing arbitrarily chosen sequence of clicks on an image without predefined click regions could not be a portable solution. Offset values are required to be initially generated and stored in the system in order to convert chosen click locations in acceptable regions to the same text password in each trial [9][10]. However, carrying these offset values to a different machine is apparently not practical. Thus, we first conduct an experiment to compare security and usability of click-based graphical passwords with and without visible grids. An earlier work [11] claimed that visible grids limit user’s freedom of choice, without conducting an experiment to justify this argument. We found that using images with visible predefined click regions improve the security and do not degrade the usability of graphical passwords. This is an important finding in its own right. In our implementation, we used images with visible grids due to the portability advantages.

The rest of our paper is organized as follows: The procedure and the design of the first experiment are detailed in Section 2. Section 3 provides the analysis of the data collected in this first experiment. Section 4 discusses other specifics in our implementation of GPEX (Graphical Password as Browser EXtension) software. Section 5 provides brief information on PwdHash, an earlier browser extension to strengthen text based passwords. Section 6 explains the methodology for our second experiment, which compares usability and security of GPEX and PwdHash. Section 7 provides the analysis of the data collected in the second experiment and Section 8 gives concluding remarks.

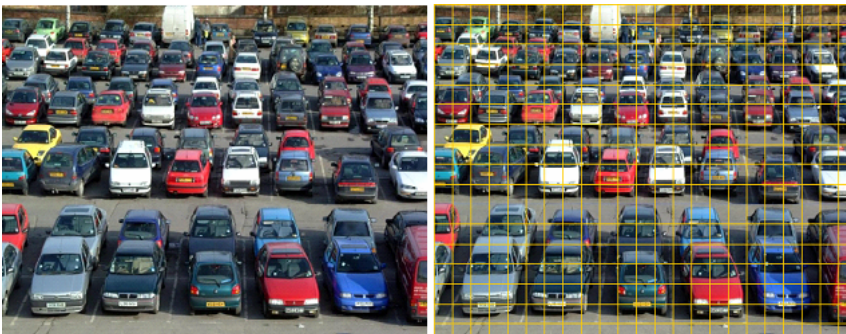
## 2 Experiment I: Click-Based Graphical Passwords with and without Visible Grids

There are numerous graphical password schemes proposed in the literature (e.g., Passpoints [11], Cued Click Points [12], and Persuasive Cued Click Points [13]) which share the same basic principle: image(s) to be clicked are displayed without any visible grids. User is free to choose his password by clicking to any point(s) on the image(s) and for a successful authentication he is expected to re-enter his password within a tolerable region centered on the original click location.

The method we have described above would have portability problems when implemented as a browser plug-in instead of the authentication method for a specific Internet application. In order to facilitate using the browser extension from any computer the user wishes to use, clicked points in a tolerable region must always be converted to the same text password since obviously the Internet application expects the same text password to be entered in the password field. Previous work [9] [10] proposed solutions to this problem for a different reason and in a different context (i.e. to be able to store the hash of the password not the password itself). However their solutions do not satisfy the portability requirement in our application scenario because of the need to pre-calculate and store offset values to translate clicks in tolerable regions to a single text based password. On the other hand, converting click locations on a pre-partitioned image to a text password is straightforward. Drawing visible (and maybe ugly) grid lines over an image is adequate to obtain a one-to-one mapping between click locations and a text based password (more detailed information on generating site-specific text-based passwords from graphical passwords will be provided in section 4).

We conducted a laboratory experiment to evaluate whether drawing grid lines over an image changes the effectiveness, efficiency, user satisfaction and security of click-based graphical passwords. This experiment as well as our second experiment was approved by the ethics committee of Middle East Technical University. Performance memory and data entry speed were compared for grid and non-grid conditions. Each participant either saw the stimuli in Figure 1a, which was first used in [11], or Figure 1b. In both cases, users click on five points as their click-based passwords.

If Figure 1b is used, participants have to choose exactly the same grids to confirm their password. Each grid is a square with 20x20 pixels in size. As a convention,



**Fig. 1.** The stimuli used in Experiment I: An image with (a) grids (right) and (b) without grids (left)

clicks on the grid line are accepted inside the below and right of that line. For Figure 1a, a tolerance square of 19 pixels centered on the original click-point is allowed to confirm the password. In both cases, participants are required to click in the correct order. The image is 451x331 pixels in size.

Effectiveness of click-based graphical passwords was measured with the number of participants who forgot their passwords, number of attempts to remember the password, and the amount of time to enter the correct password. Efficiency was measured with the amount of time to generate and confirm passwords. User satisfaction was evaluated with a questionnaire. Security was evaluated with a hot spot analysis.

Forty six subjects participated in the experiment. All of the participants were students or employees of TOBB University of Economics and Technology (average age: 22.5 years). There were twenty four males and twenty two females. Participation was voluntary. Participants were randomly allocated to one of the two experimental conditions: password image with grid ( $n=23$ , 11 males, 12 females) and without grid ( $n=23$ , 13 males, 10 females).

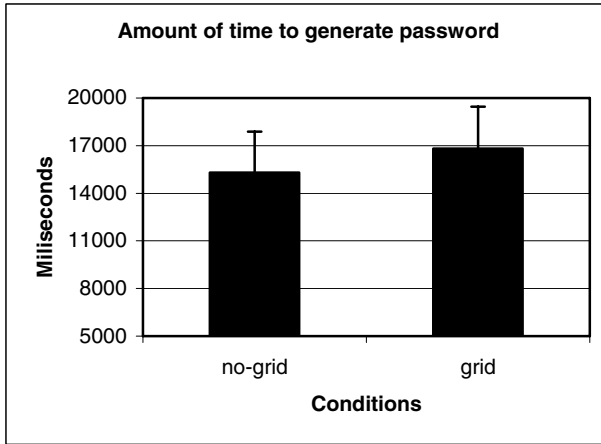
The data was collected with a desktop computer. Each participant sat facing a computer screen. Participants gave responses using mouse. They were informed about click-based graphical passwords, and they were introduced to the stand alone version of the graphical password manager system. The experiment began with a practice session in which participants practiced generating and confirming click-based graphical passwords. Passwords were generated by clicking five points on the picture and confirmed by re-clicking these points on the same order. The number of clicks to go was presented at the bottom of the image. Participants who failed to confirm their passwords repeated the practice session. The images of the practice session were different from the experiment session. Participants in the grid group practiced with images with grid lines, and participants in the non-grid group practiced images without grids.

Following the practice session, participants generated and confirmed a click-based graphical password which they had to remember later. Then participants left the laboratory. They were re-invited to the laboratory after at least four days. In the test session participants were asked to remember their click-based graphical password. Participants continued their attempts until they remembered the correct password or they decided that they could not remember the password. Participants decided themselves that they forgot the password. Finally, a questionnaire (a 5-point Likert scale) on usability of click-based graphical passwords was completed.

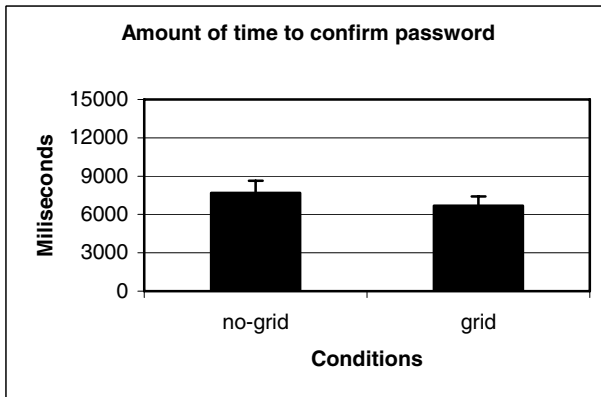
### 3 Results and Discussion of Experiment I

Effectiveness, efficiency, user satisfaction and security were investigated. Efficiency was measured with the amount of time to generate (Figure 2) and confirm password (Figure 3). There was no significant difference between grid and no-grid group for either of these measures ( $t(44) = -0.41$ ,  $p>0.05$  for generation;  $t(44) = 0.83$ ,  $p>0.05$  for confirmation).

Effectiveness of click-based graphical password was measured with the number of participants who forgot their passwords, number of attempts to remember the password, and the amount of time to enter the correctly remembered password. Four



**Fig. 2.** Amount of time to generate a click-based graphical password for grid and no-grid conditions



**Fig. 3.** Amount of time to confirm a click-based graphical password for grid and no-grid conditions

participants in the grid group (17%) and five participants in the no-grid group (22%) forgot their passwords. There was no significant difference between groups ( $X^2(1) = 0.138, p > 0.05$ ). Number of attempts to remember the correct password is presented in Figure 4, which was not significantly different between groups ( $t(35) = -0.52, p > 0.05$ ). Amount of time to enter the correctly remembered password is presented in Figure 5. There was no significant difference between groups either ( $t(35) = -0.19, p > 0.05$ ).

User satisfaction was evaluated with a questionnaire (5-point Likert). Answers were compared with non-parametric Mann-Whitney U test. None of the comparisons revealed a significant difference between grid and no-grid groups with respect to user

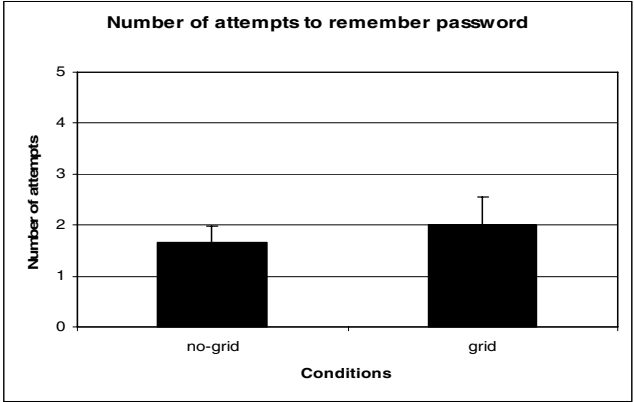


Fig. 4. Number of attempts to remember a click-based graphical password for grid and no-grid conditions

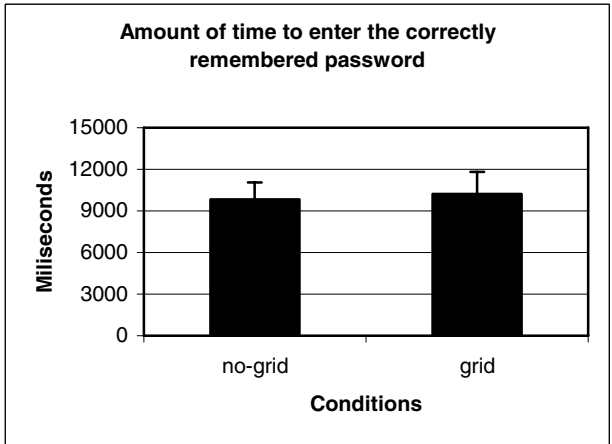


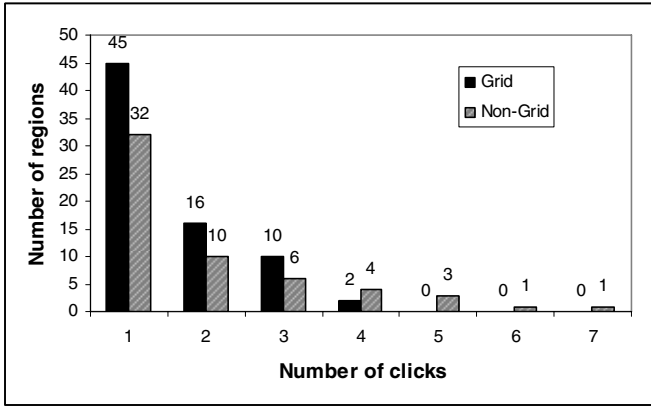
Fig. 5. Amount of time to enter the correctly remembered password

satisfaction. As a result, we can conclude that drawing grid-lines on an image did not change usability and user satisfaction of click-based passwords.

Hot-spots are regions on images that are clicked with higher probabilities compared to other regions. Hot-spots create security vulnerabilities for click-based graphical password systems since they reduce the effective size of password space. In other words, graphical passwords can be broken by less effort by conducting a hot spot attack similar to dictionary attacks possible in text-based passwords [2].

To compare the security of graphical passwords for grid and non-grid conditions, we conduct a hot-spot analysis of passwords chosen by subjects in the experiment. For each condition of Experiment I, 23 participants clicked 115 points in total.





**Fig. 6.** Number of regions versus number of clicks for grid and no-grid images

Figure 6 presents histograms showing number of regions with respect to number of times clicked by subjects in grid and non-grid conditions. For instance, in grid condition there are 45 regions clicked only one time, 16 regions clicked two times and so forth. In non-grid case, when a click is in the tolerance region of another click, we enroll them in the same region.

The method we compare these two histograms is described as follows: First of all, we devise the following formula to calculate the expected number of clicks per region.

$$E_r = \binom{115}{r} \times \left(\frac{390}{391}\right)^{115-r} \times \left(\frac{1}{391}\right)^r \times 391$$

$E_r$  denotes the number of regions chosen “r” times. Since we have a total of 115 points, the probability of choosing the given region for “r” times is  $\left(\frac{1}{391}\right)^r \times \left(\frac{390}{391}\right)^{115-r}$ . We insert  $\binom{115}{r}$  in order to consider the order of the choice.

Finally, we multiply it with total number of regions (391) in order to find the expected value.

Secondly, using this formula, we determine that if each user click is an independent random event we expect 86 regions clicked for one-time, 13 regions clicked two-times and 1 region clicked three times. There are 100 different regions in total expected to be clicked in the idealized case.

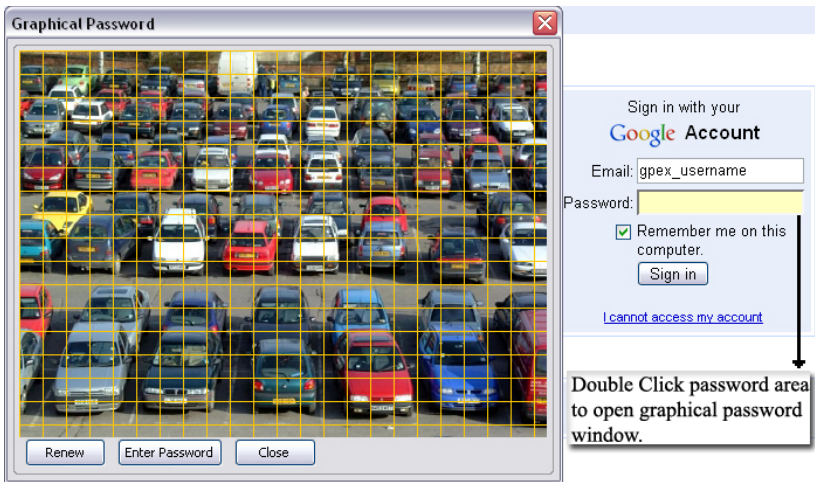
And finally, we decide whether grid or non-grid image is more vulnerable to a hot-spot analysis by comparing the deviations of the corresponding histogram from the expected values calculated. Using figure 6, we calculate that 73 and 57 different regions were clicked with grid and non-grid images, respectively. Since with non-grid image, clicks were concentrated more on specific regions, we conclude that visible

grids help to alleviate the hot-spot problem<sup>1</sup>. One explanation for this difference is that with visible grids in addition to other image features (cars, plates, colors, etc.) users can also benefit from grid lines to choose and memorize their passwords. In other words, visible grid lines help to enrich the image to select among more choices.

As a result of Experiment I, we conclude that drawing visible grid lines over an image does not affect usability and user-satisfaction of click-based graphical passwords. More than that, it helps to reduce the number of hotspots.

## 4 GPEX: Graphical Passwords as Browser Extension

In this section we introduce GPEX as a novel method to experience more secure and user-friendly web interactions. GPEX is an authentication system implemented as a plug-in to the Firefox web browser. The implementation is publicly available at <http://myuceel.etu.edu.tr/gpex>. It uses click-based passwords based on selection of at least five points on a picture which has horizontal and perpendicular grid lines with 20 pixel gaps between each adjacent parallel line. User interface of GPEX is very simple and includes a picture (451 x 331) and three buttons namely “Renew”, “Enter Password” and “Close” (Figure 7).



**Fig. 7.** The screenshot of the GPEX browser extension activated while gmail.com is visited (the box and the arrow showing it is not part of the GPEX interface)

In order to activate graphical password extension, a user should double-click the password field on a web page. If the clicked field is actually a password field, extension window pops up and the color of the password field turns to yellow indicating that a password is expected (Figure 7). Grid lines on the image constitute 391 squares

<sup>1</sup> This hot-spot analysis is our preliminary work. As a future work, we plan to collect more data and make a more elaborate investigation on this topic.

and all clicks in the same square are regarded as equivalent entries. To create a password, user should select at least 5 points by clicking inside squares. Clicking on the grid line is accepted as clicking right and/or bottom square of the line. For security reasons, less than five clicks is not accepted as a legitimate password entry.

There are three buttons on the pop-up window: renew, enter password and close (see Figure 7). The functions of the buttons are as follows:

**Renew:** If a user selects a point by mistake or decides to change the selected points he can click this button to initialize the process. After clicking this button, the system re-sets selected points and user must click all five points again.

**Enter Password:** After five points are selected and user is sure that correct points are clicked, user should click on this button. This button activates the hashing process, which will be explained shortly, and generates a unique password for the visited web site. The selected (double-clicked) password field is filled with the generated password, color of the field turns to its original color and pop-up window disappears.

**Close:** This button is used to close the window when the user decides to give up entering his password. Password field turns to its original color.

Implementation details of GPEX are summarized as follows. GPEX first detects all password fields in the visited web page and listens for double click events. When a password field is double clicked, extension window appears. If any location other than password fields is clicked, extension window does not appear. This is an effective countermeasure against so called “mock password field attack” by which an attacker obtains each pressed key in a hidden field and writes an asterisk in the mock password field.

In the extension window, the parallel lines on the 451 x 331 resolution picture forms 391 squares in total (Figure 7). When a user clicks inside a square, the extension internally records the chosen square with respect to its row and column numbers. This recording repeats for all clicks and row and column numbers are concatenated until the “Enter Password” button is clicked. The domain name is extracted from the full web site address and when “Enter Password” button is clicked, a secure hash function takes the concatenated coordinate values and the domain name of the web site as input parameters and generates a strong site-specific password like “xC4rEnjW7v”<sup>2</sup>.

There are several types of JavaScript attacks such as keyboard and mouse monitoring, domain rewriting, etc. [3] aiming to steal user’s password while it is typed or during its submission. Graphical user interface of GPEX allows a user to enter his password by mouse clicks instead of pressing keys. This means that malicious codes that listen keyboard events can not succeed in stealing the password if GPEX is preferred.

It is true that malicious JavaScript codes can also listen to mouse events. However using GPEX when a user double-clicks a password field, a pop-up window opens and the user enters his password from this new window. Up to our best knowledge, a mouse listener event implemented as JavaScript code and embedded in the HTML of the visited web page can not listen to mouse events in another window. Thus, we can

---

<sup>2</sup> We have not considered tricky implementation issues such as multiple domain names and different rules for password syntax yet. We refer interested readers to [3] for the discussion of these issues.

say that GPEX is also superior to other password manager systems with respect to its immunity against JavaScript attacks. Note that any mouse event can be observed by all the extensions installed on the browser hence all extensions installed on the browser should be trustworthy to protect the password.

## 5 PwdHash

In this section, we provide brief information on PwdHash [3,4], a browser plug-in that automatically generates strong site-specific passwords from a user's text based password and the domain name of the visited web site. After installation, in order to use PwdHash, a user should either type @@ or press F2 before entering his password. When F2 is pressed if the active field is not a password field, PwdHash warns user not to enter his password. PwdHash uses the domain name of the website as a parameter of a hash function which automatically generates a site-specific password from the typed password. Then, the system updates the target password field accordingly. By this procedure, the system provides a solution to password reuse problem and prevents phishing attacks [3, 5].

PwdHash has no visual interface and users can be confused since it is difficult to see the difference between entering password with or without PwdHash. We will discuss usability problems of PwdHash in section 7. To login using a browser without the extension, users can go to [www.PwdHash.com](http://www.PwdHash.com) and generate their site-specific passwords there.

## 6 Experiment II: Usability Comparison of GPEX and PwdHash

Before used by end users, performing a usability analysis of software products is a best practice [8]. There are numerous usability inspection methods [7] that can be conducted by experts in the field (e.g., user interface designers). Another method to validate the usability of end products is to make a usability study in a laboratory environment with participants who are potential users of the product. On this line of argument, we designed an experiment to compare the usability of graphical (GPEX) and text based (PwdHash) password manager systems. We choose PwdHash [3] instead of Password Multiplier [4] as the text-based password manager to compare with our system because an earlier work [5] has found that PwdHash is more user-friendly and perceived to be more secure. Our experimental design and methodology is similar with the authors of [5] and we compare our findings with their results in section 7.

Twenty participants (10 male and 10 female) who did not participate in Experiment I participated in the Experiment II. Participants were undergraduate and graduate students of Middle East Technical University (average age: 24.8 years). Participation was voluntary. The design of the experiment was within-subject; participants both tested with GPEX and PwdHash password manager systems. The order of tests was balanced between subjects.

At the beginning of the experiment, participants filled a questionnaire about their Internet usage and perception of web security. In this questionnaire all participants reported visiting the web daily and their initial attitude toward web security are

**Table 1.** Results represent the number of participants (out of 20) responding yes to each question

Question	Number of Users
Do you sometimes reuse passwords on different sites?	%85 (N=17)
Are you concerned about the security of passwords?	%80 (N=16)
<b>Criteria for choosing passwords:</b>	
Easy to remember	%80 (N=16)
Difficult for others to guess	%65 (N=13)
Suggested by the system	%0 (N=0)
Same as another password	%15 (N=3)
Other	%15 (N=3)
<b>Participation in online activities requiring personal or financial details:</b>	
Online purchases	%65 (N=13)
Online banking	%65 (N=13)
Online bill payments	%35 (N=7)
Other activities	%90 (N=18)

shown in Table 1. For us, the most striking result is that 80 percent of participants are concerned about the security of their passwords. This result is an indication for the need to develop new password security systems.

In the experiment, the participants completed a set of four tasks (login, migrate, update and second login) for both GPEX and PwdHash. These were same as in [5] except [5] also included the Remote Login task (login from a computer that does not have the necessary plug-in installed). We excluded the Remote Login task because this task would be very similar for GPEX and PwdHash.

Participants completed the tasks for a specific online email system (www.gmail.com). The order of the tasks was also balanced in the experiment. In the Log in task, participants have to login to the email account by using one of the plug-ins. In the Migrate Password task, participants have to login to the email account by using the given unprotected password and then they have to change the password to a protected one by using one of the plug-ins. In the Update Password task, participants have to change the password which was previously created by the plug-in. Finally, in the Second Login task, participants have to login to the email account for a second time with their changed protected password. After finishing a particular task, participants continue with other tasks without any break. After all tasks were completed, participants answered eight questions about perceived security, comfort level, ease of the task, and perceived necessity of password manager systems.

GPEX and PwdHash software were pre-installed in different notebooks and subjects performed the four tasks on these computers. Before starting the experiment, participants were given instruction sheets providing very brief information on GPEX and PwdHash. Similar to [5], users were given fixed graphical and text passwords to minimize the effect of learning new passwords on the experiment. We also asked participants to think-aloud while they were completing the tasks.

## 7 Results and Discussion of Experiment II

We classify each task performance into one of the four categories: success (completing the task in the first attempt), dangerous success (completing the task in more than one attempt which causes security exposures), failure (not completing the task) and false completion (believing the task is completed but actually it is not). Security exposures are an issue for instance when users forgot to invoke the mechanism and the raw password is sent to the server instead [5]. Apparently, completing the task in more than one attempt does not cause any security exposure with GPEX.

The results are given in Table 2. With GPEX, all 20 participants completed all of the tasks in their first attempt. With PwdHash plug-in, the success rate was below 100% for all tasks. But none of the participants failed to complete the task or made a false completion. The results indicate that participants' password usage performances are better with GPEX plug-in compared to PwdHash.

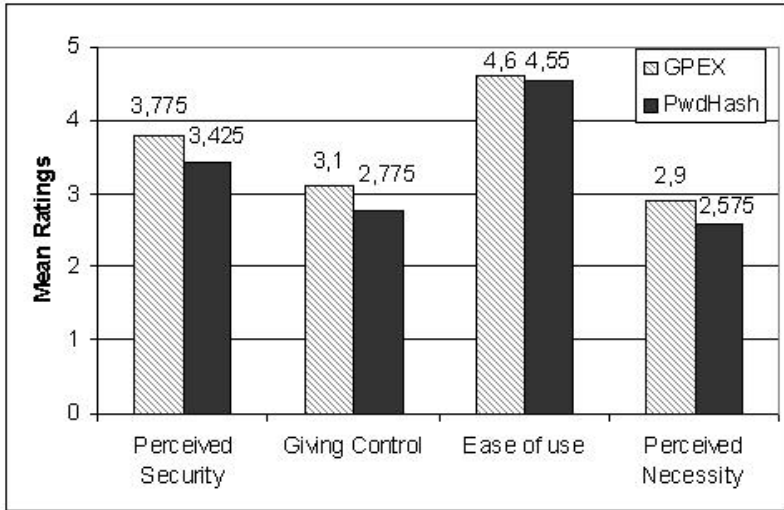
**Table 2.** Task performances for GPEX and PwdHash

Task Completion Result for GPEX				
	Success	Dangerous Success	Failure	False Completion
GPEX login	100%	N/A	0	0
GPEX migrate	100%	N/A	0	0
GPEX update	100%	N/A	0	0
GPEX second login	100%	N/A	0	0
Task Completion Result or Pwdhash				
	Success	Dangerous Success	Failure	False Completion
Pwdhash login	70%	30%	0	0
Pwdhash migrate	80%	20%	0	0
Pwdhash update	95%	5%	0	0
Pwdhash secondlogin	85%	15%	0	0

The questionnaire was as same as in [5]. Participants responded their level of agreement with various statements (see Table 3). A 5-point Likert scale was employed. Half of the questions were inverted to avoid bias. All participants completed the questionnaire for both browser extensions.

**Table 3.** Questions in the survey

<b>Perceived Security</b>
My passwords are secure when using PwdHash-GPEX.
I do not trust PwdHash-GPEX that it can protect my passwords from cybercrime.
<b>Comfort Level with Giving Control of Passwords to a Program</b>
I am comfortable with not knowing my actual passwords.
Passwords are safer when users do not know their actual values.
<b>Perceived Ease of Use</b>
PwdHash-GPEX is difficult to use.
I could easily log on to web sites and manage my passwords with PwdHash-GPEX.
<b>Perceived Necessity and Acceptance</b>
I need to use PwdHash-Gpex on my computer to protect my passwords.
My passwords are safe even without PwdHash-GPEX.



**Fig. 8.** Results of the survey

The results are shown in Figure 8. Participant believed that their passwords are secure with GPEX and PwdHash but the perceived security of two systems did not differ significantly ( $t(19) = 1.677, p = .110$ ). Both systems were perceived easy to use but there was no significant difference between them ( $t(19) = 0.335, p = .741$ ). Participants were not comfortable by giving control and there was no difference between GPEX and PwdHash ( $t(19) = 1.748, p = .097$ ). On the other hand, Participants did not find password manager systems necessary, but the perceived necessity of GPEX was significantly higher than of PwdHash ( $t(19) = 2.668, p < .05$ ).

Some results of Experiment II are in paralel with [5]. The previous work [5] concluded that major problem of PwdHash is the invisible user interface. They reported that participants did not understand how the program works and this was due to users' inaccurate or incomplete mental model of the system. With PwdHash we replicated their findings. On the other hand, with the help of visual password interface of GPEX a correct mental model of the system is easily generated and it prevents failure in login attempts.

It is important to note that GPEX is not free of usability problems, either. Some participants found it strange to click on the empty password field. Others reported that they prefer entering a password with a keyboard.

The most important difference between the results of our usability study and results of [5] is on the task completion results for PwdHash. In our experiment, the success rates for all of the four tasks were significantly higher than the rates given in [5]. The most noticeable disagreement is that while only 16% of participants completed successfully the Update Password task in [5], the ratio for the same task was 95% in our experiment. In our experiment, participants were more familiar with using the web

and they reported that they are more concerned about security of their passwords. We also observed that the participants showed great willingness to be a part of our study. However we still think that the huge difference between these two experimental results can not be explained only by change of participants and needs further research.

## 8 Conclusion

In this study we developed and tested GPEX, a password manager program implemented as a web browser plug-in to enable using graphical passwords to secure Internet applications without any need to change their authentication interface. The design of GPEX necessitates one-to-one mapping between click locations and a text based password. We achieved this by drawing grid lines over an image. First we conducted a laboratory experiment to understand whether drawing grid lines over an image changes the effectiveness, efficiency, user satisfaction and security of click-based graphical passwords. Results showed none of the variables degrade with drawing grid lines over an image. The usability comparison of GPEX with a text based password plug-in (PwdHash) showed that it is easier to use GPEX because users develop correct mental model of the system easily.

As a follow-up study, we will test usability and security of GPEX out of laboratory. We are also planning to extend GPEX by utilizing other graphical password methods.

**Acknowledgments.** We thank Osman Emre Kaya for his help on usability experiments. We thank Orhun Kara for his help on analyzing the experimental results. We also thank anonymous reviewers for their suggestions for improving the paper.

## References

1. Madigan, S.: Picture Memory. In: Yuille, J.C. (ed.) *Imagery, Memory and Cognition*, pp. 65–89. Lawrence Erlbaum Associates, NJ (1983)
2. Thorpe, J., van Oorschot, P.C.: Human-Seeded Attacks and Exploiting Hot-Spots in Graphical Passwords. In: *16th Usenix Security Symposium*, Boston, USA, pp. 103–118 (2007)
3. Ross, B., Jackson, C., Miyake, N., Boneh, D., Mitchell, J.: Stronger password authentication using browser extensions. In: *Proceedings of the 14th USENIX Security Symposium*, Baltimore, USA (2005)
4. Halderman, J., Waters, B., Felten, E.: A convenient method for securely managing passwords. In: *Proceedings of the 14th International World Wide Web Conference* (2005)
5. Chiasson, S., van Oorschot, P.C., Biddle, R.: A Usability Study and Critique of Two Password Managers. In: *15th USENIX Security Symposium 2006*, Vancouver, Canada (2006)
6. Likert, R.: A technique for the measurement of attitudes. *Arch. Psychol.* 140, 1–5.5 (1932)
7. Nielsen, J., Mack, R.L.: *Usability Inspection Methods*. John Wiley & Sons, Inc., Chichester (1994)
8. Cranor, L.F., Garfinkel, S.: *Security and Usability: Designing Systems that People Can Use*, edited collection edn. O'Reilly Media, Sebastopol (2005)



9. Bicakci, K.: Optimal Discretization for High-Entropy Graphical Passwords. In: 23rd International Symposium on Computer and Information Sciences, IEEE ISCIS 2008, Istanbul, Turkey, October 27-29 (2008)
10. Birget, J.C., Hong, D., Memon, N.: Graphical Passwords Based on Robust Discretization. *IEEE Transactions on Information Forensics and Security* 1(3), 395–399 (2006)
11. Wiedenbeck, S., Waters, J., Birget, J.C., Brodskiy, A., Memon, N.: PassPoints: Design and longitudinal evaluation of a graphical password system. *International J. of Human-Computer Studies (Special Issue on HCI Research in Privacy and Security)* 63 (2005)
12. Chiasson, S., van Oorschot, P.C., Biddle, R.: Graphical Password Authentication Using Cued Click Points. In: Biskup, J., López, J. (eds.) *ESORICS 2007*. LNCS, vol. 4734, pp. 359–374. Springer, Heidelberg (2007)
13. Chiasson, S., Forget, A., Biddle, R., van Oorschot, P.C.: Influencing Users Towards Better Passwords: Persuasive Cued Click-Points. In: *HCI 2008*, Liverpool, UK, September 1-5 (2008)

# Trust-Enhanced Recommender Systems for Efficient On-Line Collaboration

Georgios Pitsilis\*

Center of Quantifiable Quality of Service in Communication Systems, Norwegian University of Science and Technology, O.S. Bragstads plass 2E, NO-7491, Trondheim, Norway  
pitsilis@q2s.ntnu.no

**Abstract.** Trust has been explored by many researchers in the past as a solution for assisting the process of recommendation production. In this work we are examining the feasibility of building networks of trusted users using the existing evidence that would be provided by a standard recommender system. As there is lack of models today that could help in finding the relationship between trust and similarity we build our own that uses a set of empirical equations to map similarity metrics into Subjective Logic trust. In this paper we perform evaluation of the proposed model as being a part of a complete recommender system. Finally, we present the interesting results from this evaluation that shows the performance and benefits of our trust modeling technique as well as its impact on the user community as it evolves over time.

**Keywords:** Recommender Systems, Subjective Logic, Trust Evaluation.

## 1 Introduction

Recommender Systems incorporate a specific type of information filtering that has the purpose of presenting information items that are likely of interest to some user. They are widely used in e-commerce sites like *Amazon*[1] and *ebay*[2] with the aim of helping users to choose products they might like. The contribution of recommender systems comes in two forms, either as predicted ratings of services that a user wants to know about, or as lists of services that users might find of interest.

The best known technique that is used in Recommender systems is *Collaborative Filtering* CF [3]. The idea behind CF is the formation of a graph of virtual relationships that may exist between the users and is done by applying statistical techniques upon the preferences of users. The correlation of user ratings is expressed with a metric called *Similarity* and it can be calculated using the mathematical formula of *Correlation Coefficient*.

However, *Recommender Systems* and particularly CF are not perfect and because of the sparse datasets used they appear to have weaknesses such as provision of low quality predictions (known as the *false negatives* and *false positives* problems [4]), as

---

\* This work has been carried out during the tenure of an ERCIM “Alain Bensoussan” Fellowship Programme.

well as low coverage. With coverage we refer to the number of accurate predictions that can be offered to users. A prediction is expressed as a level of likeliness of a user for some particular product.

Also, the architectural characteristics of CF are known to be vulnerable to attacks from malicious and libelous users. It is the case today that recommendation provision is done by centralized entities which require access to all users ratings and preferences in order to do the appropriate correlations. Given that such data should be considered as confidential information of people it is required that such services be run by trusted authorities. Ideally such services could be run in such a way that users or entities that act on behalf of them would be able to perform the appropriate correlations and work out predictions even though they may have not been provided with full access to this information. Also, performing these correlations requires computing power which increases exponentially with the number of users and ratings [5], and that imposes a scalability problem.

Trust has been a research concept in the past as a potential solution to overcome many of the problems of recommender systems [6][7][8][9]. In our approach it is used for extending the neighboring base of users that take part in the collaborative filtering system achieving in this way the benefit of increased number of predictions that can be performed. In addition, this improvement is found to be very supportive for new users, who despite their little contribution in terms of recommendations, they can exploit the benefits of their participation early on.

This work is an extension to previous research that has been done in the past aiming to model trust for collaborative filtering systems in which trust is derived directly from user ratings [10]. In this paper we are attempting an evaluation of those modeling approaches using data from a real Recommender System. More particularly, in this work is demonstrated the benefits that users receive in terms of: a) *Accuracy* of predictions for items that users have not experienced yet and b) *Rapidity* that such information becomes available in the system for a given number of user ratings that have been gathered up to a given time. A low value is indicative of the existence of a problem which in recommender systems is known as *Cold Start Problem* [11].

## 2 Motivation

Recommender Systems (RS) are widely used nowadays and in simple terms their purpose is to suggest items, usually products, to those who might be interested in them. The main idea is to get the users that participate in such system correlated based on the opinions they have expressed in the past, with the aim to work out predictions of ratings for services or products for any interested user. The techniques used in the contemporary RS are basically based on the idea of predicted ratings being computed on ratings provided by  $k$  like-minded individuals.

RS often exist as services embedded into commercial web sites but also exist as services for supporting and providing data to researchers that are particularly interested in investigating problems of this area. *Movielens* [12], *BooksCrossing* [13] and *Netflix* [14] have been built with the sole purpose of supporting research activities. Technologies that have been applied to RS include *Nearest-neighbor* (which includes *Collaborative filtering*), *Bayesian networks* [15] and *Clustering* [16]. Bayesian

networks create a decision tree based on a set of user ratings. Despite their speed in providing recommendations they are not practical for environments in which user preferences are updated regularly. In Clustering, users are grouped by their similarity in preferences and predictions are made regarding the participation of a user in some cluster.

The basic idea behind CF is to make predictions of scores based on the heuristic that people who agreed (or disagreed) in the past are likely to agree (disagree) again. Even though such a heuristic can be sufficient to correlate numerous users with each other, systems that have employed this method still appear to be highly sparse, due to the fact that people are often unwilling to provide their feedback. As a result, systems are ineffective at making accurate predictions all the time. By *Sparsity* we mean a lack of shared experiences required for a CF system to work. The Cold start problem [4], is related to *Sparsity* and it is due to the low number of ratings that new users contribute to the system. As a result new users become isolated and hence cannot receive good quality recommendations. Apart from the Cold Start problem conventional RS face other problems such as their *Vulnerability to Attacks*.

Establishing other type of relationships including *Trust*, that could be developed between the users, especially new ones, might be helpful for increasing their contribution to the *CF* system. In this work we attempt to go one step beyond and investigate how some Trust modeling technique could outperform the traditional *CF*. More importantly we are interested to know how this benefit could become available as early as the community is still being formed and that is when the system needs it mostly. As the Cold Start problem emerges mainly during the system initialization, some demonstration of a potential solution is necessary to be accompanied by the appropriate evidence that show how the system performs over time. Therefore, it has been attempted an extensive evaluation to that captures the development of the user community.

### 3 Background Research

Trust has been proposed as a solution to alleviate the weaknesses of the standard collaborative filtering technique and various trust-based approaches of k-nearest neighbor algorithm have been introduced by many researchers in the past. The work done by Lathia et. al. [7] is focused mostly on finding the k-trusted neighbors rather than the k-similar ones to forming groups of collaborative users. Massa et. al. [6] considers the problem of receiving poor results as the inability of the recommender system to exploit other sources of the information such as the Web-of-Trust and he proposes a way of finding trustworthy recommenders via a friend-of-friend finder scheme.

For reference we mention work that has been done by other researchers in the area of *CF* to tolerate similar problems that we intended to do. In [17] Sun, Kong, Ye attempt a comparison between Person's approach, Singular Value Decomposition and Scale and Translation Invariant. In that work Pearson's approach seems to behave better during the startup phase and hence it renders more suitable for tackling the Cold Start problem. Quercia, Hailes, Carpa [18] have investigated a solution for computing trust in collaboration systems but in their proposed model the recommender's trustworthiness is not taken into account in the calculation of derived trust. In [19] there is

a technique for improving collaborative filtering based on some idea of removing global effects and in estimating the interpolation weights for each weighting factor in the Collaborative Filtering. As a result to these the estimation accuracy is improved. The work in [20] describes a framework for building hybrid CF systems which combine content and collaboration. The interesting bit of this work is the idea of setting weights on the contribution of similarity by introducing a factor which is based on the number of common items that exist in a relationship. O'Donovan and Smith [21] have introduced the idea of composing a trust value that is analogous to the percentage of accurate predictions of items in which the error is lower than a predefined threshold. In this way, the “neighbors” are filtered to the trusted ones which finally used for building up the recommendation.

To our knowledge, matters like the evolution of user communities in trust-enabled Collaborative Filtering Systems and their effectiveness against problems like the Cold-Start has not been investigated adequately so far. Instead, the main focus has been on the adaptation of trust methods onto *CF* or on the alleviation of problems mentioned above using solutions that do not involve Trust.

## 4 Description of the Proposed Idea

Our concept is based on the idea of extending the neighboring base of users by supporting the existing similarity relationships by trust relationships that can be transitively propagated throughout the network of users. In contrast to other known research approaches we used an algebra called *Subjective Logic* [22] for calculating the derived propagated trust along chains of users of known trustworthiness. In contrast to previous studies and as we are keener in capturing the benefits of using trust while a user community is being developing, in this experiment we have measured the values of the various properties of the virtual community at standard time intervals.

As it is more important to know the impact of the modeling technique as it is seen from the actual user's point of view, we attempt evaluation of a whole recommendation production cycle on which the proposed modeling has been applied on. In this way, it can be estimated the contribution of each individual mechanism (trust modeling, trust propagation etc.) to the prediction error of produced recommendations.

Next, we describe the concept of the recommendation production system we are proposing and it roughly can be considered as an extension to the existing recommendation production mechanism that is used in the *CF* systems today. In order to make easier to the reader to realize how our system can fit into an existing *RS* mechanism we provide a high level view that illustrates the individual operations.

In standard *CF* systems the correlation of user ratings is done on a nearest-neighbor basis which requires that correlated parties must have at least a minimum number of common experiences. According to this, only knowledge within a radius of one hop from a referenced node is exploitable. As will be seen later, knowledge that happens to exist at longer distances can also be made exploitable via the trust network, provided that the required mappings and transformations from trust metric to similarity can be performed.

We can imagine the entire view of the system being similar to a graph in which users are represented by vertices and the similarity relationships between them by edges.

In a simplified scenario we are supposing that users  $B$  and  $C$  have experienced some product  $k$  that user  $A$  is interested in knowing how much she might like it. In addition, users  $A$  and  $B$  could be potentially related via a similarity relationship that might exist and would normally be captured in an ordinary CF system. Moreover, we assume that user  $C$  is not related with  $A$ , as not enough evidence can be gathered to build up a similarity relationship. Considering the fact that similarity can be calculated for nodes are at distance of 1 hop from each other, similarity between nodes that are at longer distances could be derived from the trust between them, which in that case is called *indirect trust*. If we assume that trust can be transitive in long chains [23], the indirect trust can be calculated by applying the appropriate algebra on the direct trust of all entities that reside between the two nodes in the graph. Finally the derived indirect trust can then be converted into equivalent similarity. In this case, a possible lack of similarity relationships can be replaced adequately by similarities derived from Trust.

The requirement for trust to become transitive in long chains obeys that a common purpose exists along the chain. According to this, only the last trust relationship should be considered with trust for a certain purpose (functional trust) and all the other trust relationships in the chain should be with respect to the ability to recommend for the given purpose (recommender trust).

In more complex scenarios there might be multiple Trust paths going further than two hops away from the originator and finally ending to the target user which when combined together they provide a single Trust value. That Trust value which denotes how much  $A$  would trust  $C$  is again replaced by an equivalent similarity value which next is applied to Resnick’s formula [24] for computing the predicted rating.

In figure 1 below it is shown how the process of recommendation production is carried out in our system. We distinguish two main sub-paths in the rating prediction, namely the *direct* and the *indirect* one. The direct one exists only if there is direct relationship between the user  $C$ , that has already rated some item which user  $A$  is interested in, and is depicted as “Similarity  $A,C_1$ ”. That path can co-exist along with the indirect one. The indirect one requires that both  $A$  and  $C$  have established a similarity relationship with third entities  $B$  of which the direct trust for  $A$  and  $C$  can be computed. Then the hypothetical trust between  $A$  and  $C$  is derived and converted into hypothetical similarity  $A,C_2$ . Finally, the rating prediction that employs Resnick’s formula may use data derived from both paths. As an effect of this, a rating prediction can be based on a number of direct and indirect paths.

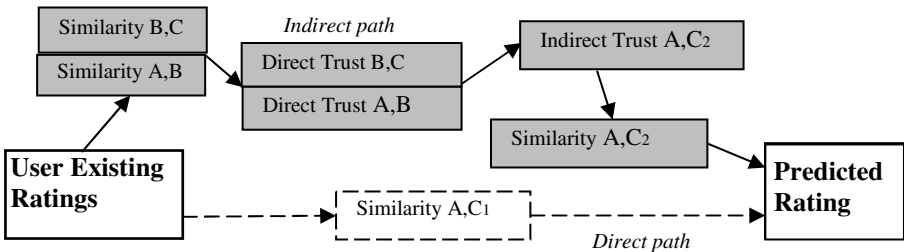


Fig. 1. The high level view of the system

There is a variety of models today for computing Trust in long chains of trustees [25][26], with various advantages and disadvantages. In our evaluation as mentioned earlier we chose *Subjective Logic* algebra to compute indirect trust using the opinions from user ratings. With *opinion* we refer to a metric of Uncertain Probabilities theory [27] which expresses the belief. Because there is always imperfect knowledge, as opinions are based on observations, lack of knowledge should be considered when assessing them. Subjective Logic framework deals with the absence of both trust and distrust by introducing the *uncertainty* property in opinions. This framework uses a simple intuitive representation of uncertain probabilities by using a three dimensional metric that comprises belief (b), disbelief (d) and uncertainty (u). which constitute an opinion. For building up opinions requires that evidence come in such a form that opinions of (b,d,u) can be derived from and thus be better manageable due to the quite flexible calculus that the opinion space provides. Unfortunately, evidence usually is available in forms that are essentially more understandable to humans. A previous work [10] we proposed for transforming data from a recommender system into the suitable format for *Subjective Logic* was necessary as this algebra can be applied onto data that come in the form of opinions. We didn't find suitable to representing opinions based on the Beta Distribution Probability Function [28] as it requires data in the evidence space to be provided strictly in binary form. More particularly, according to Beta distribution function modeling data represent two possible outcomes of a process  $X$  or  $\bar{X}$  and its behavior is described by the number of  $X$  and  $\bar{X}$  that derive from the set of observations.

As data in recommender systems are in different forms we came up with an alternative modeling solution the formulas of which we describe in the next section. In the current work we measure the impact of this modeling approach on the development of the Trust network.

As the Cold Start problem is considered as a time related issue we are mostly interested in knowing how the use of the Trust network can be exploited best so that the new users in the system can receive the benefit of their participation whenever they need it. As there are more than one candidate formulas for modeling trust from existing evidence the purpose of trying them all was two-fold. First, to identify if the use of trust in general can tolerate problems as mentioned before, and second to find the best candidate.

## 5 Experimental Evaluation

The main challenge we faced in this work was concerned with the demonstration of the evolution of the trust network which would identify the best trust modeling candidate formula for a system of which experiences grow as the time develops. In order to achieve that, it is required to be known the time at which every single recommendation has been submitted to the system, or at least the order in which all the examined recommendations have been submitted. To fulfill this requirement we used a publicly available dataset from a Movie recommender system which was provided with the time information for every rating.

As mentioned earlier data from Recommender Systems are usually available in forms not suitable to be processed by a trust algebra. In this respect, we came up with

various modeling approaches for converting the data into beliefs [10] and we finally concluded that we should perform analysis of the behavior of only 3 candidate formulas. The available data sample we used for the experiment was taken from a real collaborative filtering system for data captured during a period of 812 days. More specifically we used data from *Movielens* recommender system [12] from which we built up virtual sample communities of 100 users large. This size was chosen as being optimal for demonstrating the performance of our concept and at the same time keeping the evaluation time within satisfactory limits. Computation time is found to be an issue in such experiments, for instance we mention that in a 2.5 GHz single core CPU the computation time of a single time instance for all 5 samples of 100 users finally exceeds two days.

The 3 alternative formulas shown below were those used for shaping the belief property ( $b$ ) to be used by *Subjective Logic*, from evidence such as *user Similarity*.(shown as CC). The first formula in total was applied for 3 different values of  $k$ .

$$b = \frac{1}{2}(1-u)(1+CC^k) \quad (1)$$

Equation (1) is used for values of  $k=3, 1/3$  and 1. The  $k=1$  denotes linear transformation. CC is the similarity value (known as *Correlation Coefficient*).

$$b = \frac{1}{2} + \left( \frac{\arcsin(CC)}{\pi} \right) \cdot (1-u) \quad (2)$$

$$b = \frac{1}{2} \left( \sin\left(CC \cdot \frac{\pi}{2}\right) + 1 \right) \cdot (1-u) \quad (3)$$

For calculating the uncertainty we used the simplified formula:  $u = (n+1)^{-1}$ , in which  $n$  denotes the number of common experiences in a trust relationship between two parties A and B. For more information about the formulas used see [10]. For being able to compare the performance of the Trust enabled approach against the standard CF technique we also tried the standard CF algorithms onto the experimental data.

We performed a series of tests in which the prediction accuracy was expressed in MAE between the real rating and its predicted value (noted as recommendation) using the leave-one-out technique. Moreover, due to the unstable behavior of Pearson's similarity we considered that a similarity relationship between two users exists only if there are at least 10 common experiences.

With regard to measuring the evolution of the system we performed the experiments in time frames which differ from each other in the number of ratings that were considered for calculating a recommendation. As the number of recommendations performed at every time stage is more important to be shown than the timestamp information we considered as the best solution to present the adjacent *sparsity* value. With sparsity we refer to the percentage of empty cells in the matrix of users by items. Non-empty cell denotes existence of rating for this item from a particular user. The algorithm used for the evaluation of the total system is presented in fig.2. We call  $iTrust(i,j)$  the indirect trust between entities  $i$  and  $j$ .



<u>Let</u> $\mathbf{K}$ be the set of all users	
<u>Let</u> $\mathbf{t}$ the examined timestamp	
<u>Let</u> $\mathbf{R}$ be the set of all ratings	
<u>Let</u> $\mathbf{R}_t \subset \mathbf{R}$ be the set ratings submitted before time $\mathbf{t}$	
<u>Let</u> $\mathbf{R}_u \subset \mathbf{R}_t$ be the set of the ratings of some user $\mathbf{u}$	
<u>Let</u> $\mathbf{K}_i \subset \mathbf{K} :  \mathbf{R}_u  \geq 10$	* for users who have at least 10 ratings < $\mathbf{t}$
<u>For all</u> $i \in \mathbf{K}_i$ <u>do</u>	* for all users that belong to this category
<u>Let</u> $\mathbf{E}_i \subset \mathbf{R}_u$	* The set of ratings of $i$
<u>For all</u> $\mathbf{w} \in \mathbf{E}_i$ <u>do</u>	* for all items of user $i$
<u>Let</u> $\mathbf{M} \subset \mathbf{K}_i : \forall \mathbf{p} \in \mathbf{M}, \mathbf{E}_p \subset \mathbf{R}_t,  \mathbf{E}_i \cap \mathbf{E}_p  \geq 10 \wedge \mathbf{w} \in \mathbf{E}_p$	* users that have experienced $\mathbf{w}$
$\mathbf{P}_{set} \leftarrow \{\}$	* set of users used for rating prediction
<u>For all</u> $j \in \mathbf{M}$ <u>do</u>	* all other users
$\mathbf{SimN} \leftarrow \{\}, \mathbf{TrN} \leftarrow \{\}$	* zero counters
$\mathbf{S} \leftarrow \mathbf{CC}(i, j)$	* Pearson's similarity *
<u>If</u> ( $\mathbf{S} = \text{null}$ )	
$\mathbf{T} \leftarrow i\mathbf{Trust}(i, j)$	* Derived <i>Indirect</i> trust *
$\mathbf{TrN} \leftarrow \mathbf{TrN} + \{i\}$	
<u>If</u> ( $\mathbf{T} \neq \text{null}$ ) $\mathbf{S} \leftarrow f(\mathbf{T})$	* similarity derived from trust
<u>Else</u>	
$\mathbf{SimN} \leftarrow \mathbf{SimN} + \{i\}$	
<u>EndIf</u>	
<u>If</u> (( $\mathbf{S} \neq \text{null}$ ) $\vee$ ( $\mathbf{T} \neq \text{null}$ )) $\mathbf{P}_{set} \leftarrow \mathbf{P}_{set} + \{i\}$	* include user $i$ in rating prediction
<u>EndFor</u> $j$	
$p(\mathbf{w}) \leftarrow \text{predict}(\mathbf{w}, \mathbf{P}_{set})$	* predict value over $\mathbf{P}_{set}$
$\mathbf{MAE} \leftarrow  p(\mathbf{w}) - \mathbf{w} $	* Absolute Mean Error value *
<u>EndFor</u> $\mathbf{w}$	
<u>EndFor</u> $i$	
<u>Average</u> ( $\mathbf{MAE}$ )	
$\mathbf{TrustContr} = \frac{ \mathbf{TrN} }{ \mathbf{TrN}  +  \mathbf{SimN} }$	* trust contribution

Fig. 2. The evaluation algorithm

As it has been studied in the past [8] there is no reason for searching for trusted neighbors at distances beyond than 2 hops away from the querying node as there is no significant benefit with regard to the cost of searching. Therefore in our experiments the searches were constrained to propagate up to a max distance of 2 hops.

The sample we used comprised 73871 ratings of 500 users divided into 5 sets of 100 users. In order to study the evolution of the measured properties the ratings of each of 100 users were divided into 13 subsets based on their attached Unix type timestamp information. In this way, each subset of ratings would contain roughly the same quantity of scores that have been submitted within the same time slot.

In order to study the advantages of our system against the cold start problem we introduced the following two metrics called *System Coverage Gain* and *User Coverage Gain*, the former being a system-centric metric and the latter user-centric. The purpose of introducing them is solely to demonstrate the actual benefit that users receive when they make use of the trust graph. The metrics are computed at every timestamp TS as the system evolves. We define each metric as:

a) System Coverage Gain

This metric is characteristic to the benefits that new users receive during their early stages in using the system and SCG represents the relative benefit of the Trust-enabled method over the standard CF. In order to calculate this at every timestamp it was necessary that all rating predictions that had been produced in that particular timestamp were summed up for both the standard CF algorithm used and the Trust-enabled CF. The formula used to calculate the System Coverage Gain is:

$$SCG = \left[ \frac{prd(TR) - prd(CF)}{prd(CF)} \right]_{TS}$$

TS indicates the particular timestamp the SCG is computed for.  $prd(TR)$  is the number of predictions made by all users up to time TS if used the trust-enabled method.  $prd(CF)$  is the number of requests that would have been made by all users up to time TS if the standard CF method had been used. In this metric all ratings are considered equally the same, no matter if they come from new users or from users who have been using the system for quite a long time. Therefore SCG should be considered as the degree of opportunities that the system provides to users for making predictions for items they are interest in. In simple words, SCG expresses how the cold start problem is seen from a general point of view.

b) User Coverage Gain

Contrary to *System Coverage Gain* this metric demonstrates the benefit as it is seen from the point of view of a new user. This metric is found partially useful as this category of users is the one mostly affected by the cold start problem. On some specific timestamp TS all users who haven't supplied a single rating in the system are marked and the number of ratings they have provided is counted. The benefit every user receives on average at time TS is equal to the total number of ratings supplied by new users normalized by the actual number of new users who have encountered their first experience at that time slot. For being able to show the advantage of our technique over the standard CF, it is necessary that the above metric is been calculated for predictions achieved for both the Trust-Enabled system and a hypothetical system that employs the standard CF. The formula that is used to calculate the *User Coverage Gain* is:

$$UCG = \left[ \frac{\frac{prd(TR)}{users(TR)} - \frac{prd(CF)}{users(CF)}}{\frac{prd(CF)}{users(CF)}} \right]_{TS}$$

Where  $prd(TR)$  is the number of ratings that have been supplied by new users at the time slot that ends at TS and have made use of the trust graph for performing these

recommendations. Likewise,  $prd(CF)$  is the number of ratings that have been supplied by new users during timestamp TS and used the standard CF method. Also,  $users(TR)$  and  $users(CF)$  are the sizes of the populations of new users that have made use of the Trust-enabled technique and the standard CF respectively at time TS. In principal, all the TR-related metrics should receive higher values than the corresponded CF as the use of trust graph almost always increases the possibility for more recommendations to be produced.

## 6 Discussion of the Results

Figure 3 presents how the prediction error evolves as the time develops. In that diagram, as in all results diagrams, the time is represented by its adjacent sparsity value and it is shown across the horizontal axis. As can be seen all modeling approaches have worse performance than the standard CF almost at all timestamps. The only exception is at the first timestamp where almost all modeling approaches appear to give better results than CF. In that sense the Trust-enabled System looks less prone to the cold-start problem and thus can provide a slight benefit to the new users as the system is being built up. There is a likelihood that this benefit is being maintained for longer than the system initialization phase. For instance, in one of the 5 datasets the prediction error for all trust modeling alternatives retained lower figure than in the standard CF for the first 3 consecutive timestamps. Moreover, the penalty in accuracy for using the trust-enabled system instead of the standard CF is not very significant as its error is never higher than 2% than the error of CF. The other interesting observation is the exceptionally worse behavior of the type 2 modeling approach compared to any other approach. There is though a converging behavior with the other 4 approaches towards the end of the simulation. In conclusion, the use of trust graph does not incur

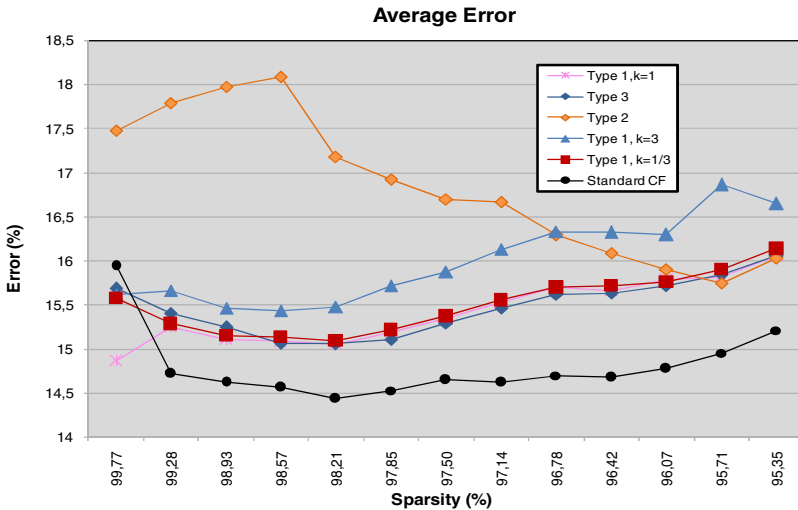
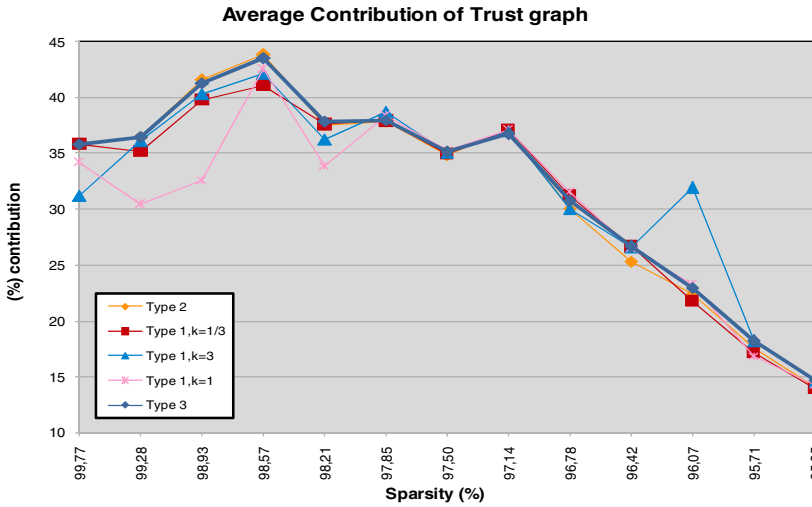


Fig. 3. Error of trust-enabled system



**Fig. 4.** Contribution of trust graph

a significant penalty in terms of accuracy, and yet more important, performs well during the beginning of the community formation.

Figure 4 shows the *Contribution of Trust Graph* for every recommendation produced. More specifically for every recommendation both the number of trusted neighbors and the total number of neighbors (trusted and similar) which have experienced the recommended item in the past are counted. We define *Contribution of Trust Graph* as the ratio of the above two values. As can be seen from the diagram, in contrast to *Prediction Error*, this metric follows a decreasing trend, but more importantly, its maximum value appears during the beginning of the community formation when sparsity is still high.

A careful examination of the correlation values of all case studies, presented in table 1, reveals the existence of high positive correlation between the *Contribution of Trust Graph* and the *Prediction Error* when the type 2 modeling formula is used. More specifically in all 5 examined user communities the correlation value appears on average to be as high as  $CC=0.84$ . That means, the more use of the trust graph is done

**Table 1.** Correlation values between Trust contribution and Prediction Error

Sample	Transformation Formula				
	Type 3	Type 2	Type 1, $k=1/3$	Type 1, $k=3$	Type 1, $k=1$
1	-0,3662	<b>0,7224</b>	-0,4275	-0,5121	-0,3389
2	-0,2633	<b>0,7393</b>	-0,3044	-0,0235	-0,1579
3	-0,5857	<b>0,6760</b>	-0,6649	-0,6114	-0,0931
4	-0,7263	<b>0,9576</b>	-0,9023	-0,7201	-0,6816
5	0,1958	<b>0,6888</b>	0,0432	0,2432	0,0291

the worse results are being received. The logical conclusion is that this modeling approach should be rejected as in the long term it does not provide any significant benefit over the other alternative approaches and according to the above evidence it is inappropriate for our work.

In all other modeling approaches the evidence shows no stochastic relationship between the error and the trust graph contribution (except for some exceptional cases) and hence there is no reason for not using them. In some exceptional dataset though (sample 4) strong negative correlation was found in the above two metrics when any modeling approach other than type 2 is used. A good interpretation of this could be: the quality of predictions is benefited by the use of trust graph. As this occurred only in one of all data sets used in the experiment we conclude that the evidence is not strong enough to support the positive claim. Hence, in practice the quality of predictions can be not affected by the decreasing use of the trust graph as the time progresses.

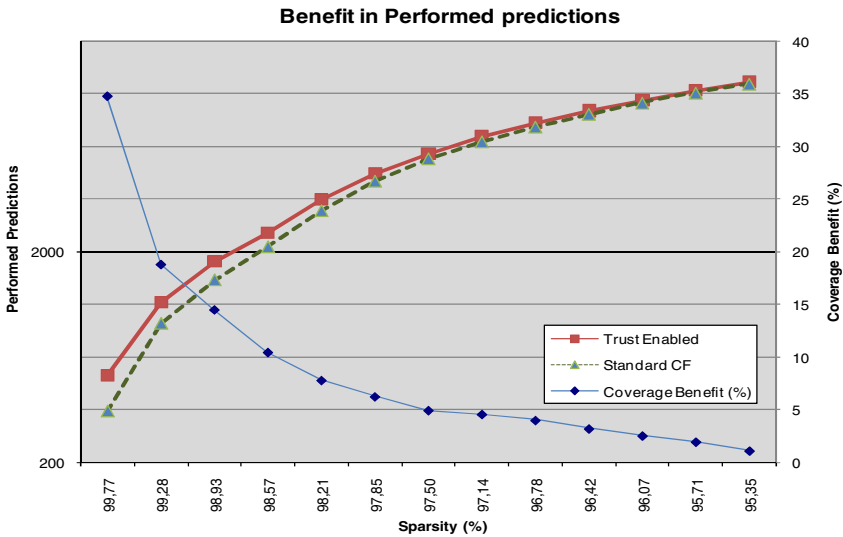


Fig. 5. The Gain and the benefit in terms of performed recommendations

In the following results we present comparisons between only one of the 3 alternative modeling techniques, which for short will be called *trust-enabled*, and the standard Collaborative Filtering technique (no use of trust graph). That is because all trust modeling techniques achieve almost the same levels of the examined metric. This comparison between the trust-enabled and the standard CF is shown pictorially in fig. 5 as figures of produced recommendations. The figure is in logarithmic scale. In continuous line is shown the recommendations that can be produced when the trust-enabled system is used and in dashed line the recommendations that can be produced if applying the standard CF. Note that in the graph the total number of recommendations

shown is that which has been provided by all users together, new and experienced ones. Hence, it renders useful as a system-centric metric.

For clarity we also present in the same figure the *Benefit* in terms of produced recommendations when the trust system is used. As can be seen, the high *Benefit* during the early stages of the system development (first timestamp) is followed by a sharp fall until the half of the evaluation time, and finally falls slowly until the end. The very low *Benefit* (lower than 5%) that is received in the second half of the simulation time is the result of saturation that occurs as more and more relationships are being established. Hence, the decreasing trend in that figure can be justified as been a consequence of the increasing number of submitted recommendations. That renders the use of trust system unnecessary as distant users are becoming reachable via similarity relationships. Finally, the *Benefit* minimizes when the similarity graph and the trust graph perfectly match. That suggests undoubtedly the use trust of graph should be made during the early stages of the system development as it is then more useful for the users.

The next two diagrams are referred to the actual gain in terms of produced recommendations. More specifically in 6 is illustrated the total number of recommendations produced by new users only, for all 5 sample communities, at each timestamp. With “new users” we refer to those who joined the system at that specific timestamp. For comparison we display the results for both the standard method and the trust-enabled system. The importance of this graph is that it focuses on the new users only and thus makes distinguishable the benefit received by users who have used the system for the first time. In contrast to the results shown in fig. 5. the user-centric view in fig. 6 shows that even though the trust-enable technique is again beneficial for new users, the actual number of recommendations produced by new users falls sharply after reaching a peak at the middle of the simulation.

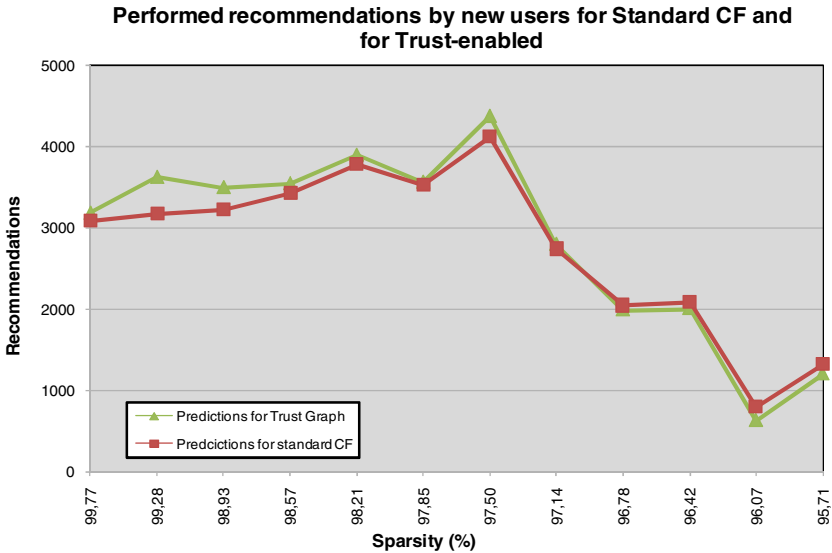
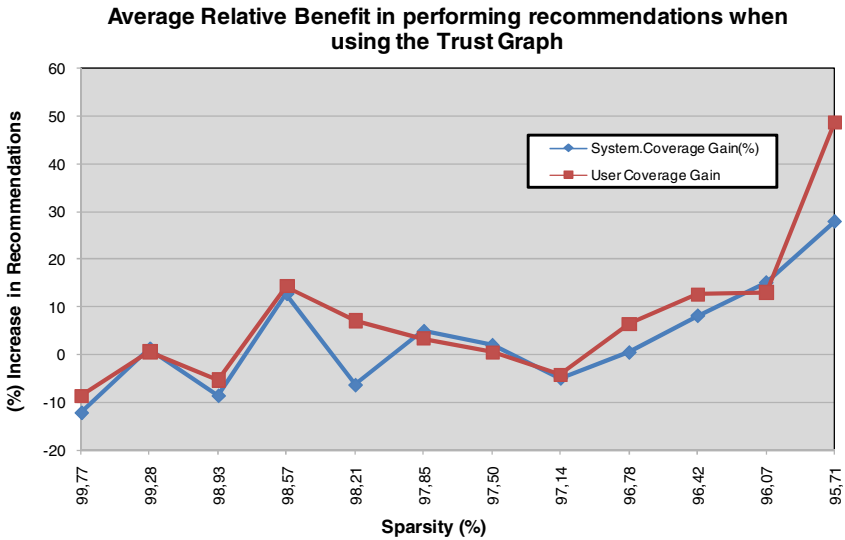


Fig. 6. Recommendations produced by new users

Another interesting observation is that during the first half of the simulation the number of recommendations, when the trust-enabled method is used, is always higher than the recommendations produced via the standard CF, which indicates quicker and more efficient user discovery in the proposed system rather than in the standard CF. However, after reaching the peak in the diagram the standard CF outperforms the trust-enabled method in terms of new recommendations. That is because the proposed system is more efficient in terms of speed at which the new users can make use of the system and thus submit recommendations. As a result, the recommendation discovery process is progressing faster than in the standard CF.



**Fig. 7.** Progress of SCG and UCG

Finally, fig. 7 shows the benefit in using the trust enabled CF expressed in metrics of *System Coverage Gain* and *User Coverage Gain* as they develop over time. The figure of UCG can be interpreted as: the benefit of new users in fact increase as the system develops as these users who join the system later actually receive more benefit than those who join early. As can be seen, both *User Coverage Gain* and *System Coverage Gain* follow nearly the same trend throughout the experiment but UCG almost at all time frames appears to be higher. Our justification is, the users on average receive more benefit than the system can observe. The negative values received for both UCG and SCG at the beginning indicate a momentary advantage of the standard CF over the trust-enabled one. In relation to diagram 6, the surprising observation is that the decreasing rate of new users' recommendations in the second half of the simulation time does not affect the average benefit they receive.

## 7 Conclusions and Future Work

We presented evaluation of a trust modeling technique with the purpose to investigate if the deployment of trust-oriented approaches could help in the alleviation of time-dependent problems. We used ratings taken from a real recommender system and we introduced metrics for expressing the benefit as seen both from the user and the system point of view.

The short experiments we performed confirm that new users do receive benefits by using the trust system, as they become more capable of performing more predictions than before. More specifically, compared to the standard CF, the method appears to provide higher potential to the actual user during the startup of the community as the prediction accuracy is maintained at very good levels. In the positive aspects we can include the faster system development as well as the fact that the quality of predictions is not affected by the decreasing use of the trust graph as the time progresses.

The increasing trend of the prediction error in the trust-enabled system as the time develops strongly suggests the use of the trust system during the startup phase, no matter which modeling approach will be chosen. As regard to the question which trust modeling formula is best for converting evidence into user opinions, the tests show there is no single formula that behaves optimally at all time instances.

As far as the benefit of the evolution test concerned, the experiment described in this paper helped very much in revealing the above findings, as static tests we performed in the past, applied on the final time instance of the same data set, had driven to very general conclusions.

The performance of CF algorithms are known to be subjective to the datasets they operate on. Therefore, the value of this work is restricted to the type of dataset used in the experiment. It would be a great advantage if tests with more datasets could be performed that would either confirm our conclusions or reveal new properties that could improve the way performance develops over time. For example, it could be investigated any likely dependency between the time step at which the traditional method outperforms the trust-enabled approach and the quality of the submitted recommendations. The objective of this would be to extend the time period during which the new users receive benefit.

For deploying such a trust-enabled system into a peer-to-peer infrastructure, in which users can join the trust community consciously, it is crucial that their participation is maintained for the longest possible period of time. Achieving this objective means getting Trust-enabled Recommender systems to work more efficiently and thus enhancing users' collaboration.

## References

1. Amazon, Electronic Commerce Company, <http://www.amazon.com>
2. Internet Company for online Auctions and shopping, <http://www.ebay.co.uk>
3. Herlocker, J., Konstan, J., Terveen, L., Riedl, J.: Evaluating collaborative filtering Recommender Systems. *ACM Transactions on Information Systems* 22, 5–53 (2004)
4. Maltz, D., Ehrlich, K.: Pointing the Way: Active Collaborative filtering. In: *Proc. of CHI 1995*, pp. 202–209. ACM Press, New York (1995)



5. Sarwar, B.M.: Sparsity, Scalability, and Distribution in Recommender Systems, Doctoral Thesis, UMI Order Number: AAI9994525. University of Minnesota (2001)
6. Massa, P., Avesani, P.: Trust-aware Collaborative Filtering for recommender Systems. In: Meersman, R., Tari, Z. (eds.) OTM 2004. LNCS, vol. 3290, pp. 492–508. Springer, Heidelberg (2004)
7. Lathia, N., Hailes, S., Carpa, L.: Trust-Based Collaborative Filtering. In: Proc. IFIPTM, Trondheim, Norway, vol. 263, pp. 119–134. Springer, Boston (2008)
8. Pitsilis, G., Marshall, L.F.: Trust as a key to improving recommendation systems. In: Herrmann, P., Issarny, V., Shiu, S.C.K. (eds.) iTrust 2005. LNCS, vol. 3477, pp. 210–223. Springer, Heidelberg (2005)
9. Papagelis, M., Plexousakis, D., Koutsouras, T.: Alleviating the Sparsity Problem of Collaborative Filtering Using Trust Inferences. In: Herrmann, P., Issarny, V., Shiu, S.C.K. (eds.) iTrust 2005. LNCS, vol. 3477, pp. 224–239. Springer, Heidelberg (2005)
10. Pitsilis, G., Marshall, L.F.: Modeling Trust for Recommender Systems Using Similarity Metrics. In: Proc. IFIPTM, Trondheim, Norway, vol. 263, pp. 103–118. Springer, Boston (2008)
11. Schein, A.I., Popescul, A., Ungar, L.H., Pennock, D.M.: Methods and Metrics for Cold-Start Recommendations. In: Proc. 25th Annual International Conference on Research and Development in Information Retrieval, pp. 253–260. ACM, New York (2002)
12. Miller, B., Albert, I., Lam, S.K., Konstan, J.A., Riedl, J.: MovieLens unplugged: experiences with an occasionally connected recommender system. In: Proc. of the 8th international conference on Intelligent User Interfaces, Miami, Florida, USA, pp. 263–266 (2003) (accepted poster)
13. Ziegler, C.N., McNee, S.M., Konstan, J.A., Lausen, G.: Improving Recommendation Lists Through Topic Diversification. In: Proc. 14th International Conference on WWW, pp. 22–32. ACM Press, New York (2005)
14. Netflix Inc., <http://www.netflixprize.com>
15. Zhan, S., Liu, L.: An Online Personalized Recommendation Model based on Bayesian Networks. In: Proc. IFIP. Research and Practical Issues of Enterprise Information Systems II, vol. 255, pp. 1575–1584. Springer, Boston (2007)
16. Breese, J.S., Heckerman, D., Kadie, C.: Empirical Analysis of Predictive Algorithms for Collaborative Filtering. In: Proc. of the 14th Conference on Uncertainty in Artificial Intelligence, pp. 43–52. Morgan-Kaufmann, San Francisco (1998)
17. Sun, X., Kong, F., Ye, S.: A Comparison of Several Algorithms for Collaborative Filtering in Startup Stage. In: Proc. Networking Sensing and Control, pp. 25–28. IEEE, Los Alamitos (2005)
18. Quercia, D., Heiles, S., Carpa, L.: B-trust: Bayesian Trust Framework for Pervasive Computing. In: Stølen, K., Winsborough, W.H., Martinelli, F., Massacci, F. (eds.) iTrust 2006. LNCS, vol. 3986, pp. 298–312. Springer, Heidelberg (2006)
19. Bell, R., Koren, Y.: Improved Neighborhood-based Collaborative Filtering. In: Proc. IEEE International Conference on Data Mining, pp. 7–14 (2007)
20. Melville, P., Mooney, R.J., Nagarajan, R.: Content-Boosted Collaborative Filtering for Improved Recommendations. In: Proc. of Eighteenth National Conf. of Artificial Intelligence, pp. 187–192 (2002) ISBN:0-262-51129-0
21. O'Donovan, J., Smyth, B.: Trust in recommender Systems. In: Proc. 10th International Conf. on Intelligent User Interfaces, San Diego, USA (2005)
22. Jøsang, A.: A Logic for Uncertain probabilities. International Journal of Uncertainty, Fuzziness & Knowledge Based Systems 9(3) (2001)

23. Cristianson, B., Harbison, W.: Why isn't Trust Transitive? In: Proc. of the International Workshop on Security Protocols, pp. 171–176. Springer, Heidelberg (1997)
24. Resnick, P., Varian, H.R.: Recommender Systems. *Communications of the ACM* 40(3), 56–58 (1997)
25. Jøsang, A., Gray, E., Kinatader, M.: Analyzing topologies of Transitive Trust. In: Proc. of the Workshop of Formal Aspects of Security and Trust (FAST 2003), Pisa (2003)
26. Rahman, A., Heiles, S.: Supporting trust in Virtual Communities. In: Proc. International Conference on System Sciences, Hawaii (2000)
27. Shafer, G.: *A Mathematical Theory of Evidence*. Princeton University Press, Princeton (1976)
28. Jøsang, A., Ismail, R.: The Beta Reputation System. In: Proc. of the 15th Conference on Electronic Commerce, Bled, Slovenia (2002)

# Towards Understanding the Requirements and Limitations of Reputation-Based Systems

Mohamed Ahmed and Stephen Hailes

University College London, UK

m.ahmed@cs.ucl.ac.uk, s.hailes@cs.ucl.ac.uk

**Abstract.** Reputation-management, as proposed for dynamic and open systems aims to provide mechanism for analysing the behaviour of agents, and to distribute this information so that the impact of the actions of those acting against the interests of a community can be limited. We study the assumptions that underpin this decision-making role for reputation-management and highlight its limitations with regard to the incentives required to realise the benefits that are claimed for it. Moreover, we show that the benefits claimed for it may not be realisable without enforcing tight constraints on the behaviour and the expectations of agents with respect to the definition of the interaction model and the incentives it presents.

## 1 Introduction

The motivation behind reputation-based trust management is straightforward: although cryptographic techniques underpin many of the available security solutions for networks, a complete reliance on such approaches to provide system security is inadvisable. In particular, such approaches: (i) have stringent management requirements, that are particularly difficult to meet in massively distributed and open environments (ii) are costly to manage (iii) lack reactivity to the behaviour of agents, relying largely on centralised rather than autonomic control, and (iv) as a consequence, are brittle when faced with uncertainty.

Reputation management aims to provide an endogenous mechanism to mediate the interactions between agents in what are typically assumed to be open and non-cooperative environments. The traditional view of trust is as something policed centrally by authorities that determine whether or not an individual is trustworthy (cf. Equifax etc.). For this to work, several premises must hold: (i) there is widespread trust in such authorities (ii) the penalties that the central authorities can impose are sufficiently severe to discourage bad behaviour (iii) it is not possible for individuals to avoid such penalties easily (for example by changing their identity). However, it is questionable whether these premises are met in existing networks and more doubtful in emerging networking environments. A more distributed approach, to complement that already in existence, necessitates (i) information collection and aggregation services to compensate for the limited local view agents have and (ii) a credible threat to discourage digressions from the standard expectations.

## 2 Related Work

There is a growing volume of work that seeks to address trust issues that result from open distributed environments. In general, such work aims to augment traditional security mechanisms with support for reactivity by using behavioural analysis techniques and ranges from intrusion detection systems [19], through to reputation-management. In the latter, the aim has been to provide two basic functions: (i) the capacity to learn (on-line) from and adapt to the behaviour of other nodes and (ii) to leverage the experiences of others, so that learning can be bootstrapped and a more accurate estimate of the behaviour of nodes under observation can be built.

Thus, for example, Ganeriwal and Srivastava [14], study the problem of generating reliable information in a sensor network and utilise a Bayesian-inspired approach to attach a subjective probability of (a measure of) integrity to the information produced by nodes. These subjective probabilities are updated by first hand experience and with the recommendations of a community of nodes. In similar fashion [6, 7, 22, 24], examine the related problems of selfish and malicious behaviour in routing of data in wireless sensor networks and mobile ad-hoc networks, and provide mechanisms for differentiating between nodes based on their observed and reported reliability. Lastly, Boukerche and Li [5] study the issues raised for reputation analysis and management when system constraints such as power and bandwidth are taken into account.

These works build on the implicit assumption that reputation-based management is a viable concept for rational agents, and that the problems lie in devising algorithms to compute the necessary trust values from the available information. There is an established body of work in the field of collaborative filtering (see [1]), but the more challenging problem is to ensure that cooperative behaviour in a collection of individual agents is a dominant strategy. For example, Levin [21] shows why simply isolating selfish nodes is insufficient to deter selfish behaviour in the case of multi-hop routing for wireless networks.

In general, with the notable exception of consideration of ephemeral identities [9, 12, 26], work that address the incentive structures or the limitation of reputation based systems [17, 23, 25] is significantly smaller in extent than the number of proposed solutions. Therefore, there is a need to study the circumstances under which the basic premises of information sharing and collective enforcement are viable and realise the dominant strategies.

This paper presents a study of some of the underlying incentive structures that influence the notion of reputation mechanisms, highlighting the need for both filtering and enforcement. Its major contribution is to provide an analysis of some the dynamics and constraints that may influence the decision to use reputation-based mechanisms to reason about the behavioural disposition of agents. We assume that the social norms required to manage and run a reputation-based mechanism are directly influenced by the prospective payoffs they entail, i.e. rational agents will only follow the rules if they are incentivised to do so (see Section 3). Given this, we show how the behavioural and structural constraints of an environment, such as the degree of observability and likelihood of repeat interaction (see Section 4), affect the incentive structure for reputation management and act either to undermine or to reinforce its viability.

### 3 Trust and Reputation

Following Conte and Paolucci [8], we will refer to the evaluations made on the basis of direct experiences as an *image* and define a *reputation* as “meta-belief” that is propagated between agents and is acquired indirectly. Instead of presenting parametric values that optimise for specific situations, for comparison we use the average discounted payoffs from following community defined social norms. To assess the credibility of the social norms, we look at the enforcement time they require in order to maintain a credible threat of a future loss in utility - relative to the loss incurred.

In the often cited definition of trust, Gambetta [13] asserts that “[Trust] (or, symmetrically, distrust) is a particular level of the subjective probability with which an agent will perform a particular action, both before [we] can monitor such action (or independently of our capacity of ever be able to monitor it) and in a context in which it affects [our] own action”. This definition informs us that trust is subjective and dependent on: (i) the competence of the agent to fulfil a task/contract, and (ii) the subjective interpretation of the result by a principal. The competence of an agent to fulfil a task is a complicated issue and may involve numerous external parameters e.g. whether the requisite resources for the task are adequate. The “subjective probabilities” that are assigned as the capacity to fulfil a task/contract depend on the strength of the incentives that drive the process of information exchange and punishment. For example, if the payoff for reneging on a contract/task is equal to the payoff gained for fulfilling it, then, without any exogenous factors, a rational agent would be indifferent to such a contact/task.

The answer to the question of “*why should an agent attempt to build up and maintain a reputation?*” must, therefore, be driven by the fact that to do so is a strategy that leads to a higher payoff than the alternatives. Kreps et al. [20] show that when there is two-sided information asymmetry about the strategy sets available to agents (i.e. when each agent has private information about its strategies), cooperation can form a sequential equilibrium in the finitely repeated prisoners dilemma. For example, by mimicking an irrational agent, e.g. playing a Tit-for-Tat strategy, rather than unconditionally defecting (as would be deduced through a backwards induction analysis), a rational agent is able to a rational principal that it is playing against an irrational agent, and that it is therefore worthwhile for the principal to forego the backwards induction argument that leads to mutual defection until towards the end of the game - in effect reaping the rewards of cooperation at the early stages of the game and a possible windfall from end-stage defection.

The *convincing* process here is the development of an image which conveys a reputation for a given characteristic. For the agent, the building and maintenance of such an image is worthwhile since being mistaken for an irrational agent leads to a higher average discounted payoff. For the principal, monitoring for and factoring a reputation into its expectation is also a worthwhile, since it is able to delay the onset of the backwards deduction argument until towards the end of the game. The significance of this work is that it shows the value in a reputation with respect to the cost of building and maintaining it.

### 3.1 The Model

To reason about the merit of a reputation-based mechanism, we situate our discussion in an environment inhabited by  $N$  interacting agents. In each time step  $t$ , a pair of agents  $(i, j)$ , each with the feasible action set  $A : \{a_1, \dots, a_n\}$  are matched with a uniform probability  $\alpha_{ij} = \frac{1}{N-1}$ , to play a symmetric stage game  $\Gamma$  and receive the payoffs  $g : A^2 \rightarrow \mathbb{R}^2$ . Being a symmetric game, the payoff matrices for the agent and principal are identical, therefore  $g(a, a') = g'(a', a)$ , whereby  $g(a, a')$  denotes the payoffs received when the principal plays  $a$  and the agent plays  $a'$ .

To reason about punishment, we follow convention and define the mini-max action  $m$ , with the payoff  $g(m, m') = \min_{a \in A} (\max_{a' \in A} g(a, a'))$  (referred to in shorthand as  $g_m$ ) and assume that  $m$  is either a pure action or an observable mixed action. Since  $m$  is not necessarily the best response to  $m'$ , for example playing a mini-max strategy in response to an opponents mini-max action may actually leave a principal worse off, we assume that  $g_m \leq 0$  [4, 18]. In this way, the mini-max action attains a negative utility and is therefore always costly. We also define  $\bar{g} = \max_{a \in A} g(a, a')$  to be the maximum possible payoff from interacting in the stage game.

The expected payoff of a principal  $i$  playing the stage game against agent  $j$  at period  $t$  is given by  $v_{(i,t)} = \alpha_{ij} g(a, a')$ . If a principal  $i$  were to receive a payoff of  $v_i$  at each time period  $t$ , with a discount rate of  $0 < \delta < 1$  to weight the value of future payoffs, i.e. a  $\delta$  of 0 signifies no expectation of future interactions, the desirability of a given payoff stream starting a time  $t_s$  and lasting for  $t_e$  periods is given by its average discounted value of:  $(1 - \delta) \sum_{t=t_s}^{t_s+t_e} \delta^t \alpha_{ij} v_{(i,t)}$ .

Finally, since each play by a principal  $i$  realises an expected payoff ( $v_i$ ) for the principal, we define  $V$  to contain the set of feasible and individually rational payoffs, such that  $v \in V > g_m$ . In essence, the set  $V$  is defined to contain values that are strictly greater than  $g_m$ , so that there is a clear incentive to abstain from being punished and pursue rewards.

### 3.2 Reputation and Perfect Information

In this section, we discuss the impact of perfect information on a society of agents that make use of images. Under the condition of perfect information, all members of the society observe the actions of each of its members and any feasible and individually rational outcome ( $v \in V | v > g_m$ ) can be enforced with the application of simple social (collective) norms [4, 10, 18]. In the section that follows, we relate this proposition to the aim of reputation-based systems (under the definition of [8]) and show that the aim of a reputation-based system can be seen as endogenously creating the effect of perfect information.

**Theorem 1.** (*Folk theorem with perfect information*) *With perfect information, any feasible and rational outcome ( $v \in V | v > g_m$ ), can be supported in the sub-game perfect equilibrium for large  $\delta$ .*

*Proof.* Suppose that play operates under a trigger-strategy based social norm, dictating that: *if no one has deviated in the last  $T$  rounds, a principal 'i' play the strategy that*

<sup>1</sup> I.e., each agent tries to minimise the maximum payoff that its game partner can receive.

yields the payoff  $v_i$ . If, however, anyone deviates, the principal switches to the mini-max strategy that yields  $g_m$ . This means that while an agent is cooperative, the whole population is cooperative towards it and each other, i.e. all agents play for the individually rational outcome  $v_i$ . However, once an agent defects, all agents must respond by defecting against it and each other (playing for  $g_m$ ) for  $t_p - 1$  periods. In effect, the social norm realises a collective punishment on the whole community for the initial digression of a defector. Under this trigger-strategy social norm, if:

An agent  $i$  does not deviate from the social norm, it can expect to attain at the minimum an average discounted payoff of  $v_i$ .

An agent  $i$  does deviate from the social norm at time  $t_d - 1$ , and it is punished for the next  $t_p - 1$  periods, it can expect to attain at most  $\delta^{t_d-1}(1-\delta)\bar{g} + vp_i$ . Where  $vp_i = (\delta^{t_d} - \delta^{t_d+t_p})g_m + \delta^{t_d+t_p}v_i$ , is the continuation payoff of the agent after the defection.

Since by definition,  $v_i > 0 \geq g_m$ , the first term of  $vp_i$ , (the punishment cost of  $(\delta^{t_d} - \delta^{t_d+t_p})g_m$ ) is negative or equal to zero, the social norm is dominant ( $v_i > vp_i$ ) iff:

$$\begin{cases} (\delta^{t_d} - \delta^{t_d+t_p})g_m < 0, \text{ or} \\ (\delta^{t_d} - \delta^{t_d+t_p})g_m = 0, \text{ and } \delta < 1, \text{ or} \\ (\delta^{t_d} - \delta^{t_d+t_p})g_m > 0, \text{ and } (1 - \delta^{t_d+t_p})v_i > (\delta^{t_d} - \delta^{t_d+t_p})g_m \end{cases} \quad (1)$$

The first two conditions of Equation 1 are given by the model assumptions, e.g.  $g_m < 0$  and  $\delta < 1$ , while the last condition simply covers the case in which  $g_m > 0$ ; insisting that if this is the case we require  $v_i > g_m$ . In effect, in the case where the punishment payoff is positive, the payoff for following the social norm *must* strictly be greater than the punishment payoff 2 to make the cooperative option attractive. Second, given a high enough discount factor to represent a patient agent, i.e.  $\lim_{\delta \rightarrow 1}$ , then  $\delta^{t_d-1}(1-\delta)\bar{g} \approx 0 < v_i$  3, and the cooperative action dominates since  $v_i > \delta^{t_d-1}(1-\delta)\bar{g} + vp_i$ .

The trigger-based social norm dictates that the digression of some agent  $j$  be punished by the whole community. To be in-line with social norm, all agents are expected to play their mini-max strategies whenever they encounter  $j$ . However, a mini-max payoff of  $g_m \leq 0$  is not necessarily justifiable for a rational agent to enforce. Therefore, without an explicit incentive to enforce the social norm, it is difficult to argue that agents will apply punishment simply for the sake of maintaining it.

To elicit the nature of the problem in more detail, let us examine it in the most extreme case. Assume that there exists an action  $a_r \in A$  with a payoff  $g_r$ , such that the payoff from playing  $a_r$  has the following property:  $v_i > g_r > g_m$  and,  $g_r = (0, 0)$  (this is analogous to a *refusal* to take part in an interaction). Since, under this condition,  $g_r$  dominates  $g_m$ , a rational agent will always prefer to receive a payoff of  $g_r$  rather than the lower  $g_m$ . The result of this assumption is that there is no credibility in the threat of punishment. A deviating agent could expect to receive a payoff of  $\delta^{t_d-1}(1-\delta)\bar{g} +$

<sup>2</sup> If  $v_i = g_m$ , then an agent is indifferent to the punishment.

<sup>3</sup> This is true at both extremes of  $\delta$ , if  $\lim_{\delta \rightarrow 0}$ , then,  $\delta^{t_d-1}(1-\delta)\bar{g} \approx 0 < v_i$  - provided that  $t_d > 2$ , which in effect implies a preference of the agent for a defection that happens in the future, relative to the stable payoff of  $v_i$ .

$(\delta^{t_d} - \delta^{t_d+t_p})g_m + \delta^{t_d+t_p}v_i$ . Knowing the dominance of  $g_r$  ( $g_r > g_m$ ) enables an agent to reason that since all other rational agents prefer  $g_r$  to  $g_m$ , if it deviates, it can receive:

- At best  $\delta^{t_d-1}(1 - \delta)\bar{g} + (\delta^{t_d} - \delta^{t_d+t_p})g_r + \delta^{t_d+t_p}v_i$ , where it is punished with refusals to interact for  $t_p$  periods - with payoff of  $g_r$  in each of those periods, after which the game returns to normal and it is able to reap the continuation payoff of  $\delta^{t_d+t_p}v_i$ .
- At worst  $\delta^{t_d-1}(1 - \delta)\bar{g}$ , where it is punished with refusals to interact for the rest of the game.

Since  $\bar{g} > v_i$ , the best case outcome reaps higher reward than the cooperative agent case, i.e.  $\delta^{t_d-1}(1 - \delta)\bar{g} + (\delta^{t_d} - \delta^{t_d+t_p})g_r + \delta^{t_d+t_p}v_i > v_i$ . However, though the worst case scenario denies an agent the average discounted continuation payoff of  $\delta^{t_d+t_p}v_i$ , whether this is damaging depends on how long an agent expects the game to last after it cheats.

In effect, provided that the remaining time in the game is less than  $\frac{\bar{g}}{v_i}$ , it is always advantageous to defect. Therefore, to create a credible threat of punishment, the act of enforcing (punishing) must itself be enforced. To reintroduce the credibility of the threat of punishment, we must state that the social norm punish digressions both in the interaction and enforcement stages of the game. Therefore, under this setting, if:

An agent  $i$  does not deviate from the social norm, and carries out  $t_p - 1$  periods of punishment as required, then the cost it incurs for enforcing the social norm is at most  $(\delta^{t_s} - \delta^{t_p})g_m$ , where  $t_s$  is the period of the first punishment. Further, since agents are matched under a uniform probability, the probability of a given agent enforcing all of  $t_p$  periods of punishment is  $(\frac{1}{N-1})^{t_p}$ , therefore the expected cost incurred from enforcing  $t_p$  periods of punishment is:  $(\frac{1}{N-1})^{t_p}(\delta^{t_s} - \delta^{t_p})g_m + [(1 - (\frac{1}{N-1})^{t_p})(\delta^{t_s} - \delta^{t_p})g_m]$ .

An agent  $i$  does deviate from the social norm, it will in turn be punished for  $t_p$  periods.

For one period of deviation at period  $t_s$  (with a payoff of  $g_r$ ), the cost incurred for the deviation is at the very least:  $\delta^{t_s}(1 - \delta)g_r + (\delta^{t_s} - \delta^{t_s+t_p})g_m$ <sup>4</sup>.

Given such payoff profiles, provided that  $N > 2$ , then the cost of not deviating is strictly less than the cost of deviating since deviating on a punishment phase results in just restarting the process. Therefore, for a rational agent, it is much more beneficial to follow the social norm. In short, under a perfect information system, where the enforcement of social norms is incentivised, the short-term gain from deviating from the norm is outweighed by the long-term loss.  $\square$

The proof in [Theorem 1](#) asserts that the social norm is sub-game perfect, because it is independent of all previous history of play, and no one has an incentive to deviate. If agents deviate during the punishment stage, their action just leads to a restarting of the punishment.

<sup>4</sup> Though of course this value is dependant on the distribution of agents that may defect on enforcing the punishment, since we cannot make any realistic assumptions about such a distribution, for now we ignore it and assume that agents are willing to enforce.



As discussed, the process of reputation management addresses the requirement on the availability of information by enabling agents to exchange their experiences of interaction. Agents may use these experiences to build up some expectation (represented as a prior) of each other, and to inform some conditional probability mechanism on what is currently known about a prospective interaction partner. To provide credibility, reputation management as often proposed (see [16] for review) makes an implicit and necessary assumption when performing an evaluation based on such a prior, namely, that it can be relied upon to act as a credible threat in deterring deviations from the norm.

If the requirements on information availability and credible threat hold, then the level of cooperation that a given agent can expect can be made directly dependent on the priors that are built about it. For example, agents whose conditional probability for cooperative behaviour leads to a lower than expected payoff, could expect to receive a relatively lower level of cooperation than would otherwise be the case.

## 4 Imperfect Information and Repeat Interactions

In a distributed computing environment, it is very difficult to sustain the assumptions that underpin the proof in [Theorem 1](#), namely the capacity of all agents to monitor all interactions in the environment. Within economic literature, the problem of imperfect information is tackled by using a type tagging mechanism [4, 10, 12, 18]. The principle is simple: if we can provide a local information processing mechanism that is honest and tamper proof -in the form of a tag, then agents can update the tags after interactions and base their decisions on the content of tag alone. However, the requirements for such a mechanism match those of a PKI and are therefore unrealistic in an open distributed system.

In the presence of imperfect information, if the opportunity for repeat interaction exists, we can still achieve a long run cooperative equilibrium without resorting to trusted third parties or an exogenous tagging scheme. The simplest non tag-based schemes are founded on bilateral trust (reciprocity) [3] and rely on repeated interactions to build histories of the outcomes of their interactions with peers. However, in distributed and open network environments, bilateral trust mechanisms face two shortcomings; (i) there may be a limited scope for repeat interaction, and hence the information accumulated in a history may not be a good approximation of the real behaviour of an agent; (ii) as we shall see, reciprocity alone may not always provide a credible deterrent.

The requirement for information sharing and collective enforcement may be characterised by looking at the standard single shot prisoners' dilemma, in which there is no information sharing and the utility preference ordering favours the non-cooperative outcome i.e. given the actions *cooperate* or *defect*,  $g(d, c) > g(c, c) > g(d, d) > g(c, d)$ . Under these limiting assumptions, it is still possible to return to a cooperative equilibrium if we can incentivise the future payoffs of an agent in a game in terms its current actions. Indeed, this is precisely the aim of an image sharing reputation system.

From the perspective of maintaining play on the equilibrium path, a limited scope for repeat interaction introduces a change in the structure of a game. Unlike the case for just imperfect information where the aim is to make up for the information deficit,

to maintain cooperative behaviour as the dominant social norm (via a credible threat of punishment), a principal has to now also be concerned with the actions of others, i.e. whether other principals in the environment will punish on its behalf. If agents are able arbitrarily to defect on the punishment stage, then the threat against a defecting agent is weakened and in a long run game, there are agents that can sustain a payoff greater than their mini-max payoff while still defecting. The awareness of this uncertainty in the effectiveness of punishments re-introduces the backwards deduction argument for rational agents and leads to a breakdown of cooperation.

For this reason, there is a two-fold aim in introducing an aggregate opinion mechanism to support the decision making: (i) to compensate for the lack of complete information by informing principals about the full behavioural scope of their prospective partners and (ii) to support an endogenous enforcement mechanism from the application of collective action; so that there is a credible threat of punishment in the form of a loss in future utility. In both scopes, a game's participants are both the witnesses and enforcers of the game. Therefore, the reliability of the mechanism as whole relies on the ability to persuade the interacting agents into enforcing the social norms of the community - which may at times conflict with their own rational interests [15].

#### 4.1 Incentives for Information Sharing

In the case where there are no universally trusted third parties and information is to be collected and distributed by the peers to whom it pertains, it is required that: (i) agents share the information that they individually collect so as to fill the information gap. (ii) This information is sufficiently trustworthy so that it is capable of influencing the decisions of others. As we shall see in this section, these two issues are not independent. To understand the incentives that guide the information sharing process, we consider the two possible cases under which information may be shared:

##### 4.1.1 Case 1: There Are No Consequences Attached to the Utterances of Witnesses

Under this assumption, when witnesses share information, they are able to make claims with the knowledge that they will not face any penalties or reprisals if the information they provide turns out to be inaccurate. This situation is analogous to cheap-talk games as discussed in detail in [11], where the content of a message does not affect the future payoff of its sender.

The consequence of this assumption is that if all message providers have the same preferences over the actions of the message receiver, i.e. all senders would prefer a receiver to act cooperatively to their messages, then, regardless of what they believe or know, they will only provide information that is good for their cause. For example, suppose we have an environment made up of  $K$  types of agents  $K : \{k_1, k_2, \dots, k_K\}$ , and the feasible sets of actions  $A : \{a_1, a_2, \dots\}$  and messages  $M : \{m_1, m_2, \dots\}$  that they may exchange. Now suppose that, there exists a pure strategy equilibrium that can be used to identify the type of a message sender based on the message they send. Under this strategy, message senders of type  $k_i$  all send message  $m_i$ , and, in turn, if a message receiver receives  $m_i$ , it will assume that the message originated with an agent of type  $k_i$  and will play action  $a_i$  in response.

Suppose also that all agents know the pure strategy equilibrium of the game, and strictly prefer that a message receiver play a specific action, say  $a_j$  with them, rather than the action ordained by their type. Under this assumption, irrespective of their actual type all agents will strictly prefer to send message  $m_j$  knowing that the receiver will assume the sender to be of type  $k_j$  and play  $a_j$  in response.

Such a uniformity of preferences over the message space (a pooling equilibrium) will lead to: (i) a reduction in the value of the message  $m_j$  for agents of type  $k_j$ , and, (ii) it will reduce considerably the capacity of message receivers to differentiate between message sender types based on the messages they receive alone. Though the existence of a pooling equilibrium reduces the applicability of cost-free messaging, a principal can still *learn* to differentiate between message sender types through repeat interaction.

To analyse the merit of learning to differentiate between message senders, assume that the learning process is on-line and is divided into two phases: (i) an initial set of interactions that may be treated as the training set, that an agent uses to learn with, followed by (ii) a latter exploitation phase, in which an agent exploits the knowledge it has learnt.

For clarity, take the simplest case in which an agent's behaviour remains constant over time, i.e., the initial training examples are able fully to describe a problem space that does not change. Assume that (i) the learning phase lasts for  $t_l$  periods and a principal incurs an average cost of  $vl$  per-period during this phase and (ii) the exploitation phase lasts for  $t_e$  periods and a principal attains an average payoff of  $ve$  per-period during this phase. Given this case, a principal  $i$ 's average discounted payoff  $v_i = (1 - \delta^{t_l})vl_i + (\delta^{t_l} - \delta^{t_l+t_e})ve_i$ , and as  $t_e \rightarrow \infty$ , we have  $\lim_{t_e \rightarrow \infty}(v_i) \approx (1 - \delta^{t_l})vl_i + \delta^{t_l}ve_i$

**Proposition 1.** *For learning to be worthwhile, a principal requires that its learning costs to be at least redeemed in its continuation payoff. More formally, it would require that:  $(1 - \delta^{t_l})vl_i \leq (\delta^{t_l} - \delta^{t_l+t_e})ve_i$ , thereby placing a constraint on the length of time that could be spent learning, or conversely the length of time the continuation period must last for the learning phase to be worthwhile. With regard to the tuples  $\{t_l, vl\}$  and  $\{t_e, ve\}$ , such that  $vl < 0 < ve$  and  $0 < \left(\frac{(1-\delta^{t_l})vl_i}{\delta^{t_l}ve_i}\right) < 1$ , we have the following lower bound for the length of the exploitation phase ( $t_e$ ):*

$$t_e \geq \frac{\log\left(1 - \frac{(1-\delta^{t_l})vl_i}{\delta^{t_l}ve_i}\right)}{\log(\delta)} \quad (2)$$

The inequality in [Equation 2](#) has the property that  $\lim_{vl_i \rightarrow 0}(t_e) \geq 0$ , therefore a smaller learning cost reduces the minimum required exploitation time for the agent<sup>5</sup>. Likewise, an agent can be seen to be more optimistic as the certainty in future interactions - a higher discount rate ( $\lim_{\delta \rightarrow 1}$ ) makes it more willing to absorb a longer training period ( $t_l$ ), or more willing to accept a larger learning cost ( $v_l$ ).

<sup>5</sup> For example, if the exploitation period lasts until the (unknown) length of a game.

<sup>6</sup> We may also hypothesise that an agent aware of a smaller learning cost may be more accepting of the risks of engaging in interactions.

The lack of repeat interaction between principals and agents or analogously the uncertainty in the identity of the prospective players, can be made represented by a lower discount rate; since both result in reducing the capacity to correlate the outcomes of previous interactions. This has the effect of increasing the lower bound of the exploitation period and amplifying the problem of learning to trust.

We can see this in another way, if the cost of learning is greater than the continuation payoff, then the numerator in Equation 2 becomes undefined, hence  $t_e$  is also undefined. On the other hand, if the continuation payoff from learning is much greater than the cost of learning, then,  $\log\left(1 - \frac{(1-\delta^t)v l_i}{\delta^t v e_i}\right) \rightarrow 1$  and  $t_e$  is not very sensitive to smaller changes in the discount rate.

#### 4.1.2 Case 2: There Is Some Consequence Attached to the Utterances of Witnesses

Under this assumption, witnesses may be held to account by some mechanism for the utterances they make. For example, if the perceived action of an agent does not correspond with the testimony/recommendation of its witnesses, then the principal may cause some loss in future payoffs for those witnesses.

However, with the assumption of imperfect information, a principal cannot know the full interaction history of a prospective partners and, consequently, the identities of other agents that the partner has interacted with in the past. In this case, when requested to provide testimony for an agent and act as a witness, agents in the environment can plausibly deny any knowledge they may have. Therefore, the option of not reporting the outcome of previous interactions is open to a witness.

For the moment, assume that a principal is able to exact punishment on a witness for a misleading recommendation by again playing the mini-max strategy  $m$  and yielding a payoff of ( $g_m \leq 0$ ) for its self and the witness. Under this scheme, we have three possible outcomes for the payoff a witness attains with regard only to the information it shares:

1. If a witness's recommendation is in agreement with the outcome of an interaction for which it has testified, then the witness's average discounted payoff with regard to the the information it has shared is either:
  - (a) 0: if the witness is neither punished or rewarded for the information it provides and, therefore its overall payoff is independent of its testimonies and dependent only on its first person interactions.
  - (b)  $\delta^{t_{rs}}(1 - \delta^{t_r})g_{rw}$ : where  $g_{rw}$  is some reward that the witness receives for  $t_r - 1$  periods starting at period  $t_{rs}$ <sup>7</sup> for the information that is has provided.
2. If the witness's testimony is judged as misleading and as a result it is punished for  $t_p - 1$  periods starting at period  $t_s$ , then its average discounted payoff with regard to the its role as a witness is at the minimum  $(\delta^{t_s} - \delta^{t_s+t_p})g_m$ <sup>8</sup>.

<sup>7</sup> In the case of a single shot reward, received at some period  $t_{rs}$ , we may substitute  $\delta^{t_{rs}}(1 - \delta^{t_r+1})g_{rw}$  with  $\delta^{t_{rs}}g_{rw}$ .

<sup>8</sup> A rational argument for the value of  $t_s$  is  $t_d + 1$ , where  $t_d$  is the period that the defection occurs. For example, given length of the game is unknown, a rational agent want to affect the punishment as soon as possible.

3. If again the witness's testimony is judged as misleading, but it is assumed to have colluded in the defection by deliberately providing a misleading recommendation, then for a defection that occurs at period  $t_d$  with a punishment phase, starting at period  $t_s$  and lasting for  $t_p - 1$  periods, the witness's average discounted payoff with regard to the its role is at a maximum  $\delta^{t_d}\bar{g} + (\delta^{t_s} - \delta^{t_s+t_p})g_m$ .

If after accepting the positive testimony of a witness on behalf of an agent, a principal is defected against by the agent, then to maintain a *credible threat* the principal must be able to exact a punishment that is at the very least equivalent to its loss in utility.

In this situation, the most limiting assumption that a principal can make with regard to the witness is that it was deliberately misled and the witness was party to a collusive act that resulted in the observed outcome. For example the witness received some part of the utility lost<sup>9</sup>. Under this assumption, given the following global preference ordering of the possible payoffs,  $\bar{g} > g_{rw} \geq 0 \geq g_m$ :

**Proposition 2.** *If no rewards are received by the witness for providing information, then to maintain a credible threat, the principal would require that:  $0 \geq \delta^{t_d}\bar{g} + (\delta^{t_s} - \delta^{t_s+t_p})g_m$ , or simply, the cost of the punishment phase must outweigh the payoff from defecting,  $\|(\delta^{t_s} - \delta^{t_s+t_p})g_m\| \geq \delta^{t_d}\bar{g}$ , which, from the perspective of length of the punishment periods ( $t_p$ ) requires:*

$$t_p \geq \frac{\log\left(1 - \frac{\delta^{t_d}\bar{g}}{\delta^{t_s}g_m}\right)}{\log(\delta)} \quad (3)$$

**Proposition 3.** *If rewards are received for information, the length of the reward period  $t_r$  is conditional on the start of the reward period. To ensure that receiving the reward and then not cheating is a dominant strategy, we must maintain the following inequality  $\delta^{t_{rs}}(1 - \delta^{t_{rs}+t_r})g_{rw} \geq \delta^{t_d}\bar{g} + (\delta^{t_s} - \delta^{t_s+t_p})g_m$ , which from the perspective of the size of the reward payoff gr requires:*

$$g_{rw} \leq \frac{\delta^{t_d}\bar{g} + (\delta^{t_s} - \delta^{t_s+t_p})g_m}{\delta^{t_{rs}}(1 - \delta^{t_{rs}+t_r})} \quad (4)$$

Or, which from the perspective of a punishment, requires that the cost of punishment be less than the gain from receiving a reward, then deviating, i.e.  $(\delta^{t_s} - \delta^{t_s+t_p})g_m \leq -\delta^{t_{rs}}(1 - \delta^{t_r})g_{rw} - \delta^{t_d}\bar{g}$ <sup>10</sup>. Therefore the length of the punishment period required to maintain a credible threat is:

$$t_p \geq \frac{\log\left(1 - \frac{-\delta^{t_{rs}}(1 - \delta^{t_r})g_{rw} - \delta^{t_d}\bar{g}}{\delta^{t_s}g_m}\right)}{\log(\delta)} \quad (5)$$

Given the threat of a loss of future payoff, we can see that a witnesses' disposition to share information is directly conditional on how that information will be interpreted by the receiver.

<sup>9</sup> This particular example ignores the situation that the witness has been misled into providing a misleading testimony.

<sup>10</sup> This is equivalent to stating that absolute value of the punishment cost is greater or equal to the continuation payoff for defecting, i.e.  $\|(\delta^{t_s} - \delta^{t_s+t_p})g_m\| \geq \|-\delta^{t_{rs}}(1 - \delta^{t_r})g_{rw} - \delta^{t_d}\bar{g}\|$ .

## 4.2 Incentives for Collective Punishment

So far, we have discussed the incentives for agents to support a cooperative community by contributing to reduce an information deficit. In this section we examine the other half of the problem: their incentives to act to deter what is classified as socially detrimental behaviour by a community of agents, i.e. to punish /enforce social norms on behalf of a community.

To do, so we will look at three related mechanisms to support collective action for the purpose of making the threat of punishment credible. Depending on the degree imperfect information and the likelihood of repeat interaction, we have the following mechanisms available; (i) collective punishment, (ii) personal punishment and (iii) personal enforcement.

### 4.2.1 Collective Punishment

Collective punishment is the process in which the whole community of agents reacts to a defection and all players are punished during the punishment period. As a social norm, it is typically presented in situations where there is maximal imperfect information and little or no opportunity for repeat interaction. Group sanctioning based social norms aim to reduce the information and state requirements of a system, and look to identify the types of cooperative equilibria that can exist in the most limiting circumstances, i.e. the contagious equilibrium.

These mechanisms are built on “public grim trigger strategies” (PGTS) [10, 18], in which the social norms dictate that agents play for  $v_i$ , so long as they have not been defected against, and switch to  $g_m$  either forever or for  $t_p$  periods (when combined with a public forgiveness mechanism) if they have been defected against. It should be obvious from the mechanism described for PGTS based social norms that, once defection occurs, it starts to spread epidemically.

Now, given the PGTS social norm which prescribes  $t_p - 1$  periods of punishment for a defection that occurs in period  $t_d$ , and define  $\gamma_{ij}$  to be the probability of an agent  $i$  being matched to play a defecting agent  $j$ :

- If no agent defects, the average discounted payoff is  $v_i$ .
- If no agent has defected so far, and agent  $i$  defects, then the average discounted payoff is at most  $(1 - \delta^{t_d})\bar{g} + (\delta^{t_d} - \delta^{t_d+t_p})[\gamma_{ij}g_m + (1 - \gamma_{ij})\bar{g}]$ .
- Once a defection has occurred, the average discounted payoff during the punishment phase for an agent that chooses to defect is at most:  $(1 - \delta^{t_p})[\gamma_{ij}g_m + (1 - \gamma_{ij})\bar{g}]$ .
- The average discounted payoff for a cooperative agent during the punishment phase is at least:  $(1 - \delta)\underline{g} + (\delta - \delta^{t_p})[\gamma_{ij}g_m + (1 - \gamma_{ij})v_i]$ .

For the threat of punishment to be credible, we require the following payoff ordering  $a > b > c > d$ . This inequality is sustained by the both the diminishing returns offered by discounting future payoffs ( $\delta$ ) and epidemic growth rate of  $\gamma_{ij}$  -the probability of interacting with a defecting agent, once defection starts.

As a realistic and feasible strategy, the PGTS strategy is too fragile, it is indiscriminate in its punishment, too easily susceptible to noise or to deliberate manipulation by a malicious agent seeking to destroy a cooperative equilibrium and, finally, it cannot

always converge back to the cooperative equilibrium<sup>11</sup>. Personal punishment schemes are proposed to overcome these issues.

### 4.2.2 Personal Punishment

Personal punishment is the process in which only the defector is punished by the group [4] and is typically presented in situations in which there is some public information, but again little to no opportunity for repeat interaction. Personal punishment schemes aim to overcome the main fragility of a community enforcement mechanism. However, there are a number of constraints attached to such mechanisms.

First given that some form of collective punishment is required to maintain a credible threat against defection when there is a limited opportunity for repeat interactions. To focus only on the incentive issues for enforcement, assume that (i) defections during the interaction phase can be monitored, but defection at the punishment stage cannot, and, (ii) the social norm prescribes that all agents that defect on either on the interaction or punishment stage be punished for  $t_p$  rounds.

Given this scenario, if an agent defects against a principal at period  $t_d$ , and is punished for  $t_p - 1$  periods starting at  $t_s$ , then average discounted value of the payoff stream during this phase is  $\delta^{t_d} \bar{g} + (\delta^{t_s} - \delta^{t_s+t_p}) g_m + \delta^{t_s+t_p} v_i$ , while for cooperating, it expects a payoff of  $v_i$ . Since the punishment phase payoff is necessarily less than the payoff in a cooperative phase, to maintain a credible threat of punishment, a rational agent ought strictly to prefer to cooperate rather than to defect.

In this situation, looking at the continuation payoff alone is somewhat misleading. The problem becomes clearer if we look at situations in which there exists an asymmetry in the feasible payoff sets available to agents; for example, if different types of agents attain differing payoffs from the same outcome. According to Bó [4] the threat of personal punishment incentivised by the long term continuation payoff cannot be a credible threat in this situation.

To overcome this problem, we must change our initial assumption by monitoring and/or incentivising the enforcement behaviour of agents, or modify the payoff mechanism. Otherwise, if there is significant imperfect information, and we are led back to the PGTS as the only sub-game perfect equilibrium.

To address this problem, both [4, 12] present mechanisms that work by modifying the payoff landscape. Both schemes introduce short term incentives for enforcers. In the case of Bó [4], these are in the form of “forgiveness” offerings, i.e.  $\exists a_f \in A$ , such that  $g(m, a_f) > g_m \geq g(a_f, m)$ . By playing  $a_f$ , the agent being punished asks for forgiveness (with the threat of being mini-maxed if it does not) and the principal gains an incentive to enforce the social norm in the form of the payoff from playing  $g(m, a_f)$ .

In a similar fashion, Friedman and Resnick [12], use a variation of the PGTS named “paying your dues” (PYD), in which instead of the threat of unyielding punishment, agents can instead be indebted to the interaction process through an incentive scheme that rewards cooperative behaviour rather than punishing deviant behaviour.

<sup>11</sup> Strictly speaking, this is not the always true, PGTS based mechanism can be tuned to perform to certain conditions, and even die out, however, this is still fragile and difficult to control [2].

Therefore, given a fee of  $g_f$ , and an agent's individually rational payoff of  $v_i$ , for the fee paying scheme to be a dominant strategy, it is required that rational agent at least be able to recoup it in some 't' future interactions. Therefore given a cost of entry of  $g_f$ , iff:

An agent  $i$  does not deviate from the norm, u the most desirable outcome is that; after paying the fee the agent proceeds to be cooperative for a game that lasts for  $t_e - 1$  periods. In this case, average discounted payoff is at least  $g_f + \delta(1 - \delta^{t_e-1})v_i$ .

An agent  $i$  does deviate from the norm at period  $t_d$  - having already paid the entree fee, given a punishment of  $t_p - 1$  periods starting at  $t_s$ , the average discounted payoff is at least  $g_f + \delta^{t_d}\bar{g} + (\delta^{t_s} - \delta^{t_s+t_p})g_m + (\delta^{t_s+t_p} - \delta^{t_s+t_p+t_e})v_i$ .

**Proposition 4.** *To enable the loss of the entry cost to act as a credible threat of punishment, we require that:*

$$g_f + (\delta - \delta^{t_e})v_i \geq g_f + \delta^{t_d}\bar{g} + (\delta^{t_s} - \delta^{t_s+t_p})g_m + (\delta^{t_s+t_p} - \delta^{t_s+t_p+t_e})v_i \quad (6)$$

*Necessitating the following constraints; first, given  $0 < \delta < 1$  and  $v_i > 0$ , the lower bound of the length of the exploitation period ( $t_e$ ) is:*

$$t_e \geq \frac{\log\left(\frac{\delta^{t_d}\bar{g} + (\delta^{t_s} - \delta^{t_s+t_p})g_m + \delta^{t_s+t_p}v_i - \delta v_i}{(\delta^{t_s+t_p} - 1)v_i}\right)}{\log(\delta)} \quad (7)$$

*Second with regard to the credibility of the punishment payoff, given  $0 < \delta < 1$  and  $v_i > 0$ , we require that:*

$$(\delta^{t_s} - \delta^{t_s+t_p})g_m \leq -g_f - \delta^{t_d}\bar{g} - (\delta^{t_s+t_p} - \delta^{t_s+t_p+t_e})v_i \quad (8)$$

*Therefore the lower bound on the length of a punishment period ( $t_p$ ), is:*

$$t_p \geq \frac{\log\left(\frac{-g_f - \delta^{t_d}\bar{g} - \delta^{t_s}g_m}{\delta^{t_s}(v_i - \delta^{t_e}v_i - g_m)}\right)}{\log(\delta)} \quad (9)$$

Unless these bounds can be satisfied, no rational agent has an incentive to pay a joining fee of  $g_f$ , because a rational agent would expect that either it cannot be policed or exploited relative to the cost of the investment.

Notice that in the PYD scheme, there exists either some universally trusted mechanism to verify the age of identities (the likelihood of repeat interaction under same identity), or individual repeat interaction can be relied upon to satisfy the requirements on  $t$  as shown. If there is no mechanism for verifying the age of identities, and a limited opportunity for repeat interaction, then the PYD mechanism loses its incentive appeal. In short, in the personal punishment scheme long-term continuation payoffs cannot be relied upon when there exists asymmetry in the payoff structures, therefore the act of punishing (enforcement) must also be incentivised.



### 4.2.3 Personal Enforcement

Personal enforcement in which only the agents that have been defected against carry out the punishment. This type of social norm is typically presented in situations in which reciprocity can be applied, i.e. regardless of the level of imperfect information, there is significant opportunity for repeat interaction. Under this assumption, reciprocity is feasible, and bilateral trust mechanisms easily apply. Axelrod [3], provides comprehensive work on the feasibility and credibility of reciprocity. However, the assumption of repeat interactions narrows the applicability of reciprocity in this case.

## 5 Conclusion

We have highlighted the need for both information sharing and collective sanctioning to support reputation-management and discussed the problem of incentivising them. To be incentivised, both of these features require commitments, relative to the costs they incur and seem infeasible if such commitments cannot be fulfilled.

From a design perspective, we may use these results to reason about the applicability and requirements of reputation based systems, especially when the intended deployment environment requires a strong level of assurance, or when trying to understand how to bootstrap such a system. Further, if we can specify the requirements for reputation based systems, for example, the associated payoffs, we have the opportunity to treat the problem from a mechanism design perspective and to engineer more direct solutions that reflect the associated payoff structures, the degree of observability and likelihood of repeat interaction, rather than rely on the more vague incentive mechanism offered by information sharing and classification alone.

We have also shown that there is tangible value in both information of high quality and commitments to collective action; expressed as a factor of the payoffs realised from utilising them. This raises the possibility of using markets to manage both information sharing and collective action, i.e. the development of agents that trade information and act to police an environment for example through applying enforcement on behalf of their clients. We plan to explore these possibilities in future work.

**Acknowledgement.** The authors would like to thank the EC for funding under the ARAGORN project and the anonymous reviewers for their helpful comments.

## References

1. Adomavicius, G., Tuzhilin, E.: Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering* 17, 734–749 (2005)
2. Ahmed, M., Hailes, S.: Controlling contagious equilibrium. Technical report, Department of Computer Science, University College London (2009)
3. Axelrod, R.: *The Evolution of Cooperation*. Basic Books, New York (1984)
4. Bó, P.: Social norms, cooperation and inequality. *Journal of Economic Theory* 30(1), 89–105 (2007)
5. Boukerche, A., Li, X.: An agent-based trust and reputation management scheme for wireless sensor networks. In: *IEEE GLOBECOM* (2005)

6. Buchegger, S., Le Boudec, J.Y.: The effect of rumor spreading in reputation systems for mobile ad-hoc networks. In: Proceedings of WiOpt 2003: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks, Sophia-Antipolis, France (March 2003)
7. Buchegger, S., Le Boudec, J.Y.: A robust reputation system for peer-to-peer and mobile ad-hoc networks. In: Proceedings of P2PEcon 2004, June 2004. Harvard University, Cambridge (2004)
8. Conte, R., Paolucci, M.: Reputation in Artificial Societies: Social Beliefs for Social Order. Kluwer Academic Publishers, Dordrecht (2002)
9. Douceur, J.R.: The sybil attack. In: Druschel, P., Kaashoek, M.F., Rowstron, A. (eds.) IPTPS 2002. LNCS, vol. 2429, pp. 251–260. Springer, Heidelberg (2002)
10. Ellison, G.: Cooperation in the Prisoner's Dilemma with Anonymous Random Matching. Review of Economic Studies 61(3), 567–588 (1994)
11. Farrel, J., Rabin, M.: Cheap talk. Journal of Economic Perspectives 10, 103–118 (1996)
12. Friedman, E., Resnick, P.: The social cost of cheap pseudonyms. Journal of Economics and Management Strategy 10(2), 173–199 (2001)
13. Gambetta, D.: Can we trust trust? In: Gambetta, D. (ed.) Trust: Making and Breaking Cooperative Relations, electronic edn., ch. 4, pp. 49–72. Department of Sociology, University of Oxford, 1988/2000 (1988)
14. Ganeriwal, S., Srivastava, M.B.: Reputation-based framework for high integrity sensor networks. In: Proceedings of SASN 2004, Washington, DC, USA (October 2004)
15. Gibbons, R.: Trust in Social Structure: Hobbes and Coase Meet Repeated Games. In: Cook, K. (ed.) Trust in Society. Russel Sage Foundation, Thousand Oaks (2000)
16. Jøsang, A., Ismail, R., Boyd, C.: A survey of trust and reputation systems for online service provision. Decision Support Systems 43(2), 618–644 (2007)
17. Jurca, R., Faltings, B.: An incentive compatible reputation mechanism. In: Proceedings of the IEEE Conference on E-Commerce, pp. 285–292 (2003)
18. Kandori, M.: Social Norms and Community Enforcement. Review of Economic Studies 59(1), 63–80 (1992)
19. Kim, J., Bentley, P., Wallenta, C., Ahmed, M., Hailes, S.: Danger is Ubiquitous: Detecting Malicious Activities in Sensor Networks using the Dendritic Cell Algorithm. In: Bersini, H., Carneiro, J. (eds.) ICARIS 2006. LNCS, vol. 4163, pp. 390–403. Springer, Heidelberg (2006)
20. Kreps, D., Milgrom, P., Roberts, J., Wilson, R.: Rational Cooperation in the Finitely Repeated Prisoners Dilemma. Journal of Economic Theory 27, 245–252 (1982)
21. Levin, D.: Punishment in Selfish Wireless Networks: A Game Theoretic Analysis. In: First Workshop on the Economics of Networked Systems (2006)
22. Michiardi, P., Molva, R.: Core: A collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. In: Communication and Multimedia Security (September 2002)
23. Miller, N., Resnick, P., Zeckhauser, R.: Eliciting honest feedback in electronic markets. Technical report, SG Working Paper Series RWP02-039 (2002)
24. Quercia, D., Lad, M., Hailes, S., Capra, L., Bhatti, S.: STRUDEL: Supporting Trust in the Dynamic Establishment of peering coalitions. In: Proceedings of the 21st ACM Symposium on Applied Computing, Dijon, France (April 2006)
25. Luke Teacy, W.T., Patel, J., Jennings, N.R., Luck, M.: Travos: Trust and reputation in the context of inaccurate information sources. Journal of Autonomous Agents and Multi-Agent Systems 12 (2006)
26. Yu, H., Kaminsky, M., Gibbons, P.B., Flaxman, A.: Sybilguard: Defending against sybil attacks via social networks. In: ACM SIGCOMM (2006)

## Appendix

### Proof of [Proposition 1](#)

Given that the learning period and exploitation period are distinct and exploitation starts directly after learning. We may characterise principal  $i$ 's average discounted payoff as:

$$(1 - \delta) \left[ \sum_{t=0}^{t_l-1} \delta^t v l_i + \sum_{t=t_l}^{t_l+t_e-1} \delta^t v e_i \right] \quad (10)$$

$$= (1 - \delta^{t_l}) v l_i + (\delta^{t_l} - \delta^{t_l+t_e}) v e_i$$

Now, if learning is to be worthwhile for a principal, it is required that  $(\delta^{t_l} - \delta^{t_l+t_e}) v e_i \geq (1 - \delta^{t_l}) v l_i$ . Given  $0 < \delta < 1$ , solving [Equation 10](#), the lower bound of  $t_e$  is:

$$t_e \geq \frac{\log \left( 1 - \frac{(1 - \delta^{t_l}) v l_i}{\delta^{t_l} v e_i} \right)}{\log(\delta)} \quad (11)$$

### Proof of [Proposition 3](#)

The most limiting assumption that can be made by a principal that is defected against, after it has paid a reward for some information, is that the witness it used has colluded against it and, it has received two sets of rewards; first, for the information it has provided, then from the defection. Under this assumption, the average discounted payoff of the witness is:

$$(1 - \delta) \left[ \sum_{t=t_{rs}}^{t_{rs}+t_r-1} \delta^t g_{rw} + \sum_{t=t_d}^{t_d} \delta^t \bar{g} \right] \quad (12)$$

$$= \delta^{t_{rs}} (1 - \delta^{t_r}) g_{rw} + \delta^{t_d} \bar{g}$$

Now, to provide a credible threat of punishment, it is required that the punishment costs remove any gains from the actions; therefore, given  $0 < \delta < 1$  and  $0 < g_m$ , we therefore require that:

$$(\delta^{t_s} - \delta^{t_s+t_p}) g_m \leq -\delta^{t_{rs}} (1 - \delta^{t_r}) g_{rw} - \delta^{t_d} \bar{g} \quad (13)$$

Solving [Equation 13](#), the lower bound of  $t_p$  is:

$$t_p \geq \frac{\log \left( 1 - \frac{-\delta^{t_{rs}} (1 - \delta^{t_r}) g_{rw} - \delta^{t_d} \bar{g}}{\delta^{t_s} g_m} \right)}{\log(\delta)} \quad (14)$$

**Proof of Proposition 4**

First, for the cooperative outcome to dominate, we must ensure the continuation payoff of the cooperative outcome dominates the payoff from defecting, and then reaping a continuation payoff after punishment, giving us:

$$g_f + (\delta - \delta^{t_e})v_i \geq g_f + \delta^{t_d}\bar{g} + (\delta^{t_s} - \delta^{t_s+t_p})g_m + (\delta^{t_s+t_p} - \delta^{t_s+t_p+t_e})v_i \quad (15)$$

Since by definition  $v_i < \bar{g}$ , we must ensure that the difference is made up for by the cooperative continuation payoff, hence, given  $0 < \delta < 1$  and  $v_i > 0$ , from solving [Equation 15](#) the lower bound of  $t_e$  is:

$$t_e \geq \frac{\log\left(\frac{\delta^{t_d}\bar{g} + (\delta^{t_s} - \delta^{t_s+t_p})g_m + \delta^{t_s+t_p}v_i - \delta v_i}{(\delta^{t_s+t_p} - 1)v_i}\right)}{\log(\delta)} \quad (16)$$

Second, to function as a credible threat, the punishment phase must result in an average discounted payoff that is less than the cost of entry plus the gain from interactions. If this requirement does not hold, then the continuation payoff after defecting will outweigh the punishment, and will therefore diminish the threat. Therefore, we have:

$$(\delta^{t_s} - \delta^{t_s+t_p})g_m \leq -g_f - \delta^{t_d}\bar{g} - (\delta^{t_s+t_p} - \delta^{t_s+t_p+t_e})v_i \quad (17)$$

Again, given  $0 < \delta < 1$ ,  $g_m < 0$  and  $\bar{g} > v_i > 0$ , the lower bound of  $t_p$  from [Equation 17](#) is:

$$t_p \geq \frac{\log\left(\frac{-g_f - \delta^{t_d}\bar{g} - \delta^{t_s}g_m}{\delta^{t_s}(v_i - \delta^{t_e}v_i - g_m)}\right)}{\log(\delta)}. \quad (18)$$

# Elimination of Subjectivity from Trust Recommendation

Omar Hasan<sup>1</sup>, Lionel Brunie<sup>1</sup>, Jean-Marc Pierson<sup>2</sup>, and Elisa Bertino<sup>3</sup>

<sup>1</sup> INSA Lyon, France

{omar.hasan,lionel.brunie}@insa-lyon.fr

<sup>2</sup> IRIT, France

pierson@irit.fr

<sup>3</sup> Purdue University, IN, USA

bertino@cs.purdue.edu

**Abstract.** In many distributed applications, a party who wishes to make a transaction requires that it has a certain level of trust in the other party. It is frequently the case that the parties are unknown to each other and thus share no pre-existing trust. Trust-based systems enable users to establish trust in unknown users through trust recommendation from known users. For example, Bob may choose to trust an unknown user Carol when he receives a recommendation from his friend Alice that Carol's trustworthiness is 0.8 on the interval [0,1].

In this paper we highlight the problem that when a trust value is recommended by one user to another it may lose its real meaning due to subjectivity. Bob may regard 0.8 as a very high value of trust but it is possible that Alice perceived this same value as only average. We present a solution for the elimination of subjectivity from trust recommendation. We run experiments to compare our subjectivity-eliminated trust recommendation method with the unmodified method. In a random graph based web of trust with high subjectivity, it is observed that the novel method can give better results up to 95% of the time.

## 1 Introduction

Trust is an indispensable requirement for the successful operation of a number of distributed applications. Trust is defined as “the degree to which one party has confidence in another within the context of a given purpose or decision” [1]. On eCommerce websites, a buyer must trust the seller to deliver the services or goods that are promised. In ad hoc networks, a node trusts neighboring nodes to route its messages. In peer-to-peer file sharing networks, a peer trusts others to deliver authentic content. Internet forums and online communities trust members not to post spam. Without a system in place that enables users to establish the trustworthiness of other parties, a distributed application would suffer from exploitation and eventually fail to provide adequate service.

A variety of trust-based systems [2], [3], [4], [5] have been developed that enable agents (any entity capable of making trust related decisions) to determine

if the party they wish to transact with is trustworthy. Trust recommendation is a key technique that is utilized in trust-based systems for an agent to determine the trustworthiness of an unknown party. A trust recommendation is an attestation of the trustworthiness of an agent Carol by Alice to Bob, where Bob is an agent who is not acquainted with Carol but maintains a trust relationship with Alice.

We present the argument that trust evaluation by each individual is subjective and thus when two individuals exchange a trust value its meaning is distorted due to differences in their perception. For example, Alice may have suggested to Bob that the trustworthiness of Carol is 0.8 on the interval  $[0,1]$ , which according to her subjective opinion may have been average trustworthiness. However, it is possible that Bob has a different perspective on trust values and regards 0.8 as a very high value. Thus subjectivity prevents the true meaning of Alice's recommendation from being conveyed to Bob.

We subscribe to the definition of subjectivity given by the Merriam-Webster online dictionary ([merriam-webster.com](http://merriam-webster.com)) as a judgment that is "modified or affected by personal views, experience, or background" and is "peculiar to a particular individual". Several works [5], [6], [7] propose trust models that aim to capture the subjectivity aspect of human trust. However, the focus is on enabling agents to form trust opinions that are uniquely their own in contrast to delegating trust formation to some external authority. None of the cited works address subjectivity as it affects trust recommendation. In this paper we focus specifically on the problem of subjectivity in trust recommendation.

The remainder of the paper is organized as follows: Section 2 further describes the problem and discusses the notion of disposition to trust. Section 3 presents a basic trust model that serves as a framework for the development of the solution and experiments. In Section 4 we introduce and build the method for elimination of subjectivity from trust recommendation. Experiments in Section 5 that evaluate the effectiveness of the method are followed by a discussion and proposals for future work in Section 6. In Section 7, we present concluding remarks.

## 2 Background

### 2.1 Trust Representation and Subjectivity

How does one represent the amount of trust that one individual associates with another? A common approach is to represent the spectrum of trust quantitatively as a numerical range. Marsh's formalism [8] represents trust as a continuous variable over an interval of  $[-1,1]$ . Golbeck's FilmTrust [9] defines an integer range of 1 to 10. Gambetta [10], Griffiths [6], and Toivonen [11] utilize an interval of  $[0,1]$  for the purpose.

An alternate approach is to divide the span of trust into strata and assign them qualitative labels. The stratification used by Abdul-Rahman and Hailes [2] is given as the set {Very Trustworthy, Trustworthy, Untrustworthy, Very Untrustworthy}. Jonker and Treur [12] use a similar stratification defined as the ordering: Unconditional Distrust < Conditional Distrust < Conditional Trust

< Unconditional Trust. Levien’s Advogato [13] allows users to rate each other as an Apprentice (minimum trust), Journeyer (medium trust), or as a Master (maximum trust).

Let’s consider a scenario where Alice assigns a trust value of 0.8 to Carol on an interval of  $[0,1]$  with 1 representing maximum trust. Let’s assume that 0.8 is an average trust value if it is viewed in the context of trust values that Alice has assigned to other entities in the past. Thus Alice perceives Carol as someone being moderately trustworthy. With whatever skew Alice assigns trust values to other entities, it presents no problem inside her local environment since all those values lie in the same context.

The problem of subjectivity arises when Alice conveys to Bob that her trust in Carol is represented by the value 0.8. It is likely that a value of 0.8 signifies something very different to Bob. Is 0.8 an average value of trust for Bob as was the case for Alice? Or is 0.8 a very high value of trust for Bob? Given the context of Bob’s history of trust value assignments, we may discover that Bob rarely ever assigns a value of 0.8 to any entity and thus associates very high trust with such a value. In Alice’s position Bob might have assigned a value such as 0.6 to Carol. Bob may make a misjudgment of Carol’s trustworthiness if he bases his decision on his own perception of the trust value conveyed to him by Alice. We observe that due to subjectivity, the meaning of a trust value is distorted when it is propagated from one individual to another. Subjectivity occurs due to differences in the dispositions to trust of individuals. Disposition to trust is defined and discussed in the next section.

The use of strata with qualitative labels may initially be considered as a solution to the problem of subjectivity. We may argue that a stratified trust representation model, such as the four distinct strata defined by Abdul-Rahman and Hailes [2], provides clear semantics and avoids the ambiguity associated with numerical values. The reasoning being that a qualitative label such as “trustworthy” should hold the same meaning for one entity as it does for another.

However, we concur with Griffiths [6] and Marsh [8] that the stratification approach also suffers from the problem of subjectivity. Different entities may associate the same experiences with different strata. For example, based on their own perception of trust, what is viewed by Alice as “very trustworthy” may be judged as merely “trustworthy” by Bob.

We note that subjectivity, as we describe it, is not an issue for the trust representation model used by some popular commercial websites, such as Epinions (epinions.com). This is due to the fact that the resolution they provide for evaluating users is minimal. Epinions allows users to only either “Block” (not trust) or “Trust” other users. This model relies more on the quantity of ratings received per user rather than the degree of trustworthiness specified in an individual rating. On eBay (ebay.com), which uses a somewhat similar model, users value each other’s trustworthiness in the same stratum (that is “positive”) over 99% of the time [14]. Our work addresses systems that employ broader ranges for the expression of trust.

## 2.2 Disposition to Trust

Disposition to trust is the inherent propensity of an individual to trust or distrust others. An individual's disposition to trust does not vary for specific entities but is a stable characteristic of their personality that governs how they view the trustworthiness of every other entity that they encounter.

McKnight et al [15] define disposition to trust as the "extent to which a person displays a tendency to be willing to depend on others across a broad spectrum of situations and persons".

Rotter [16], [17] notes that an individual's "generalized attitude" towards trust is a product of life experiences, such as interactions with parents, peers, and authorities. Boone and Holmes [18] suggest that good experiences lead to a greater disposition to trust and vice versa.

A study in the context of ecommerce by McCord and Ratnasingam [19] has demonstrated that there is a strong relationship between an individual's disposition to trust and the trust related decisions that they make.

A thorough treatment of the literature on disposition to trust is provided by Kaluscha [20].

We now revisit Alice, Bob and Carol from our previous example. Alice and Bob are two individuals with different dispositions to trust. Alice has a high disposition to trust and thus assigns a high trust value of 0.8 to Carol. In contrast, Bob who has a lower disposition to trust, rates Carol's trustworthiness as only 0.6. This subjectivity occurs despite the fact that Carol exhibits the same behavior in her interactions with both Alice and Bob.

## 3 Trust Model

In this section we define a trust model. An important constituent of the model is the provision for trust recommendation and propagation. The objective is not to define a novel trust model but to establish a basic one that will serve as a framework within which we will develop and test our method for elimination of subjectivity from trust recommendation.

We define  $A$  as a set of agents.  $A = \{a_0, a_1, \dots, a_n\}$ . We define a binary relation  $T$  on the set  $A$ .  $T$  is a subset of  $A \times A$ .  $T = \{(u, v) : u, v \in A\}$ . The relation  $T$  represents the *trusts* relation between two agents. We will use the notation  $uTv$ ,  $u$  trusts  $v$ , and  $(u, v)$  interchangeably. In our model, the properties of the *trusts* relation are as follows:

**Property 1.** *The relation  $T$  is reflexive.  $uTu$ . An agent trusts itself.*

**Property 2.** *The relation  $T$  is not symmetric.  $uTv \not\Rightarrow vTu$ . If agent  $u$  trusts agent  $v$  then this does not imply that  $v$  also trusts  $u$ .*

**Property 3.** *The relation  $T$  is not transitive.  $a_0Ta_1 \wedge a_1Ta_2 \not\Rightarrow a_0Ta_2$ . If agent  $a_0$  trusts agent  $a_1$  who in turn trusts agent  $a_2$ , then this does not imply that  $a_0$  also trusts  $a_2$ .  $a_0$  may trust  $a_2$  or it may not.*



We define a Web of Trust as a weighted directed graph  $G = (A, T)$ . The agents in the set  $A$  form the vertices of the graph. The trust relations between agents given as ordered pairs in the set  $T$  are the edges of the graph. Since  $G$  is a directed graph, an edge  $(u, v)$  is incident from  $u$  and incident to  $v$ .

A weight is associated with every edge  $(u, v)$  in the graph, which represents the amount of trust that agent  $u$  holds for agent  $v$ . The weight associated with an edge  $(u, v)$  is given as the function  $t(u, v)$ .  $t : T \rightarrow X$ . The set  $X$  is defined as  $X = [0, 1]$ .

The range of  $t(u, v)$  is real numbers bounded by 0 and 1. 0 implies “minimum trust” and 1 implies “maximum trust”. Real numbers between 0 and 1 give us infinite resolution for expressing trust.

The absence of  $(u, v)$  in  $T$  implies that no trust relationship exists between agents  $u$  and  $v$ . We do not address distrust in this model.

A path  $p(a_0, a_k)$  of length  $k$  from an agent  $a_0$  to an agent  $a_k$  is a sequence  $\langle a_0, a_1, a_2, \dots, a_k \rangle$  of agents such that  $(a_{i-1}, a_i) \in T$  for  $i = 1, 2, \dots, k$ .

### 3.1 Trust Recommendation and Propagation

If  $(a_0, a_1) \in T \wedge (a_1, a_2) \in T$ , then  $t(a_1, a_2)$  may be considered as a recommendation from  $a_1$  to  $a_0$ . That is, taking into consideration  $t(a_0, a_1)$  and  $t(a_1, a_2)$ ,  $a_0$  may choose to establish  $(a_0, a_2)$  and  $t(a_0, a_2)$ . We say that the trust of  $a_1$  in  $a_2$  is propagated to  $a_0$ .

To facilitate the discussion we establish the following terminology:

**Source agent** – the agent from whom the path originates; the agent that may establish trust in a previously unknown agent based on the given recommendations

**Recommender agent** – an agent that recommends another agent

**Target agent** – the agent at whom the path terminates; the agent whom the source agent may choose to trust

In the preceding case,  $a_0$  is the source agent,  $a_1$  a recommender agent, and  $a_2$  the target agent.

We stress that since trust is not transitive in our model, the propagated trust is only a suggestion to the source agent regarding the trustworthiness of the target agent. The source agent may or may not choose to establish a trust belief based on this suggestion.

We generalize the notion of trust recommendation and propagation for a path of length  $k$ :

If  $(a_0, a_1), (a_1, a_2), (a_2, a_3), \dots, (a_{k-2}, a_{k-1}), (a_{k-1}, a_k) \in T$ , then  $t(a_{k-1}, a_k)$  may be considered as a recommendation from  $a_{k-1}$  to  $a_{k-2}$ ,  $t(a_{k-2}, a_{k-1})$  as a recommendation from  $a_{k-2}$  to  $a_{k-3}, \dots$ , and  $t(a_1, a_2)$  as a recommendation from  $a_1$  to  $a_0$ . Taking into consideration  $t(a_0, a_1), t(a_1, a_2), t(a_2, a_3), \dots, t(a_{k-2}, a_{k-1}), t(a_{k-1}, a_k)$ ,  $a_0$  may choose to establish  $(a_0, a_k)$  and  $t(a_0, a_k)$ . We say that the trust of  $a_{k-1}$  in  $a_k$  is propagated to  $a_0$ .

According to the classification introduced by Ziegler and Lausen [21], the trust metric presented in this section may be categorized as local and scalar. The model discussed here shares similarities with those defined by Golbeck et al [22], Chen and Yeager [23], and Abdul-Rahman and Hailes [24].

## 4 A Method for Elimination of Subjectivity from Trust Recommendation

In this section we introduce our method for the elimination of subjectivity from trust recommendation.

### 4.1 Quantitative Representation of an Agent's Disposition to Trust

The method requires quantitative representation of the disposition to trust of agents. We discuss three possible alternatives for this purpose.

**Manually specified by the agent.** The agent may be presented with a scale, for example, 1 to 10 or  $[0,1]$  and asked to rate their disposition to trust manually. The approach is simple and straightforward. However, the disadvantage of this approach is that the agent has to be explicitly engaged by the process. Moreover, it is debatable if an agent himself is a true judge of his own disposition to trust.

**Assessed through a trust scale.** A number of researchers have developed trust scales that help assess the disposition to trust of a person. The subject is required to respond to a series of questions with weighted multiple choice answers. The cumulative score of the subject indicates their disposition to trust.

Rotter's Interpersonal Trust Scale [17] and Christie and Geis's Machiavellianism Scale [25] are examples of this approach. A sample question from Rotter's Interpersonal Trust Scale is as follows:

“In dealing with strangers one is better off to be cautious until they have provided evidence that they are trustworthy.”

*Answer choices:* strongly agree (weight: 1), mildly agree (2), agree and disagree equally (3), mildly disagree (4), strongly disagree (weight: 5).

Rotter's and the Machiavellianism trust scales are likely to assess the disposition to trust of an individual accurately. However, the requirement that each agent make themselves available for a series of questions discounts their practicality.

**Inferred from an agent's history of trust value assignments.** Several examples from the computer science literature may be cited where historical patterns are used to predict future behavior with considerable success. Instances include Self-Customizing Software [26] or Adaptive User Interfaces [27], and Branch Predictors in Microprocessors [28].

We propose an approach based on similar lines for determining the disposition to trust of an agent. The trust values that an agent has assigned in the past may be considered as an indication of their disposition to trust. For example, given an agent who has a pattern of assigning high values of trust, we may infer that the agent has a high disposition to trust, and vice versa. We thus propose to represent an agent's disposition to trust by the collection of their previous trust value assignments in a system.

A close approximation of an agent's disposition to trust is possible only if they have made a significant number of trust value assignments in the past. The question is what number can be considered as significant. We experiment with multiple values in Section 5.

The primary reason we choose this approach for the representation of disposition to trust is that it does not require additional input from an agent. Given a web of trust, we can test our method without requiring each agent to explicitly establish their disposition to trust.

## 4.2 The Method

As we have discussed earlier, the trust values assigned by an agent are subjective to their disposition to trust. When a recommender agent recommends a target, the meaning of the associated trust value is distorted due to the different disposition to trust of the source agent.

The solution we propose is to report trust not as an absolute score but a value that is relative to the disposition to trust of the recommender agent. In other words, we report the relative standing of the recommender agent's trust in the target agent in terms of the trust value assignments that the recommender agent has made in the past.

Two simple options for implementing this idea are reporting trust as either a standard score (*z*-score), or as a percentile. We opt for a solution based on percentiles and not one based on standard scores since the latter requires that the trust values assigned by agents be normally distributed.

A percentile value indicates the recommender agent's perception of the target agent in relation to the others that the recommender agent has rated in the past.

Going back to the example discussed in Section 2 if Alice conveys to Bob an absolute value such as 0.8, Bob does not know if according to Alice the value 0.8 is an average value or a very high value of trust. However, if the trust is reported as a percentile value, Bob does have this information. For example, if the percentile value is in the vicinity of 50%, Bob would know that according to Alice, Carol has an average trustworthiness. If the percentile value is around 80% or 90%, it is clear that Alice regards Carol as highly trustworthy. The absolute value that Alice locally assigned to Carol becomes irrelevant.

To convert the percentile to a local absolute score the source agent reads the value that is at the given percentile in the collection of trust values that he

himself has assigned to other agents. This absolute score holds perfect meaning for the source agent since it is in the context of his own disposition to trust.

Thus going through a relative value as an intermediary, the subjectivity and misinterpretation associated with an absolute trust value are eliminated.

We note that this method does not require agents to make any modifications to the way they evaluate other agents. Locally, each agent establishes their trust beliefs as usual, in terms of their own disposition to trust. Another positive aspect of this solution is that it does not require the involvement of any third parties and is therefore suitable for decentralized networks.

### 4.3 Formal Description of the Method

Within the framework of the trust model discussed in Section 3, a formal description of the method follows.

$d_u$  is a collection of the weights associated with the outgoing edges of agent  $u$ , that is, all  $t(u, v)$  where  $v$  is a node adjacent to  $u$ . As discussed in Section 4.1, the collection of trust values previously assigned or  $d_u$  represents the disposition to trust of agent  $u$ .

The values in  $d_u$  are arranged in ascending order and indexed  $1, 2, \dots, n_u$ , where  $n_u$  is the number of outgoing edges of agent  $u$  (as well as the number of values in  $d_u$ ). The  $j^{\text{th}}$  value in  $d_u$  is referred to by  $d_u[j]$ . We define a function  $first(x, d_u)$  that returns the index of the first occurrence of a value  $x$  present in  $d_u$ .

$c(u, v)$  is the percentile of  $t(u, v)$  in  $d_u$ . The function which calculates  $c(u, v)$  is given as:

$$\begin{aligned} c(u, v) &= \text{percentile}(t(u, v), d_u) \\ &= \frac{100 \cdot \text{first}(t(u, v), d_u)}{n_u + 1} \end{aligned}$$

As an example, consider  $d_{Alice} = \langle 0.4, 0.4, 0.5, 0.6, 0.8, 0.8, 0.8, 0.8, 0.8, 0.9, 0.9 \rangle$  and  $t(Alice, Carol) = 0.8$ . Then  $n_u = 11$  and  $first(t(Alice, Carol), d_{Alice}) = 5$ .  $c(Alice, Carol)$  is calculated as follows:

$$\begin{aligned} c(Alice, Carol) &= \text{percentile}(t(Alice, Carol), d_{Alice}) \\ &= \frac{100 \cdot \text{first}(t(Alice, Carol), d_{Alice})}{n_{Alice} + 1} \\ &= \frac{100 \cdot 5}{11 + 1} = 41.67\text{percentile} \end{aligned}$$

$t(u, v)_w$  is defined as the value in  $d_w$  at the  $c(u, v)^{\text{th}}$  percentile. The function which calculates  $t(u, v)_w$  is stated as:

$$\begin{aligned}
t(u, v)_w &= \text{trustvalue}(c(u, v), d_w) \\
&= \begin{cases} d_w[i] + f \cdot (d_w[i + 1] - d_w[i]) & \text{if } 0 < i < n_w \\ d_w[1] & \text{if } i = 0 \\ d_w[n_w] & \text{if } i = n_w \end{cases}
\end{aligned}$$

where,

$$i = \left\lfloor \frac{c(u, v) \cdot (n_w + 1)}{100} \right\rfloor$$

and,

$$f = \frac{c(u, v) \cdot (n_w + 1)}{100} - i$$

$i$  is an integer and  $f$  is a fraction greater than or equal to 0 and less than 1.

We may think of  $t(u, v)_w$  as the value  $t(u, v)$  transformed such that instead of being in reference to the disposition to trust of agent  $u$ , it is now in reference to the disposition to trust of agent  $w$ .

Instead of reporting  $t(u, v)$ , an agent  $u$  calculates  $c(u, v)$  and communicates this percentile value to agent  $w$ . Given  $c(u, v)$ , agent  $w$  determines  $t(u, v)_w$  and considers that as the recommended value.

Continuing the example from above, consider  $d_{Bob} = \langle 0.2, 0.3, 0.3, 0.3, 0.5, 0.5, 0.5, 0.6, 0.8 \rangle$ . Then:

$$\begin{aligned}
t(\text{Alice}, \text{Carol})_{Bob} &= d_{Bob}[i] + f \cdot (d_{Bob}[i + 1] - d_{Bob}[i]) \\
&= d_{Bob}[4] + 0.17 \cdot (d_{Bob}[5] - d_{Bob}[4]) \\
&= 0.3 + 0.17 \cdot (0.5 - 0.3) = 0.33
\end{aligned}$$

where,

$$\begin{aligned}
i &= \left\lfloor \frac{c(\text{Alice}, \text{Carol}) \cdot (n_{Bob} + 1)}{100} \right\rfloor \\
&= \left\lfloor \frac{41.67 \cdot (9 + 1)}{100} \right\rfloor = 4
\end{aligned}$$

and,

$$\begin{aligned}
f &= \frac{c(\text{Alice}, \text{Carol}) \cdot (n_{Bob} + 1)}{100} - i \\
&= \frac{41.67 \cdot (9 + 1)}{100} - 4 = 0.17
\end{aligned}$$

The implementation of the functions *percentile* and *trustvalue* is based on the method for estimation of percentiles given by NIST [\[29\]](#).

## 5 Experiments

### 5.1 Experiment Design

Our objective is to test if the trust values recommended through the subjectivity-eliminated trust recommendation method are of higher quality than those given by the unmodified trust recommendation method in which trust values are conveyed without any alteration. The quality of a recommended trust value is stated as its closeness to the trust value that the source agent would assign to the target agent if it had direct experience with it.

Given a web of trust, we find paths of length 2 such that there also exists a direct edge from the source agent to the target agent. For such an instance, not only can we calculate the subjectivity-eliminated recommended trust value but we also know what value the source agent has assigned to the target agent based on direct experience. We therefore have a reference value with which we can compare the values given by the subjectivity-eliminated trust recommendation method and the unmodified trust recommendation method.

If the value given by the subjectivity-eliminated trust recommendation method is closer to the reference value than the one given by the unmodified trust recommendation method, we consider the experiment run as a success (hit) for our method. If the opposite is true, we consider it a failure (miss). If both values are the same or are within a small range (0.05) of each other, we count neither a hit nor a miss.

To facilitate the discussion we establish the following terminology:

- $\alpha$  – recommended trust value given by the unmodified trust recommendation method which does not take subjectivity into account
- $\beta$  – recommended trust value derived from the subjectivity-eliminated trust recommendation method
- $\gamma$  – trust value depicting the source agent’s trust in the target agent based on direct experience

Given  $G$ , a web of trust, and  $z$ , the minimum number of outgoing edges for source and recommender agents, the experiment is algorithmically described in Figure 1.

As discussed in Section 4.1, an agent must have made a significant number of trust value assignments in the past for a close approximation of their disposition to trust.  $z$  represents this number. We experiment with different values in Section 5.3.

Given a large and diverse web of trust we can assume that there will be both hits and misses. If the number of hits is significantly larger than the number of misses, we have an indication that the method is effective. On the contrary if the number of misses is considerably greater than the number of hits or if there is no significant pattern then we may infer that the method is ineffective.

The experiment has been implemented using the Java Graph library (JGraphT). When determining an alternate path, the first path returned by Dijkstra’s algorithm that meets the given criteria is used. In the following sections we describe a web of trust and proceed with experiment runs.

SUBJECTIVITY-EXPERIMENT( $G, z$ )

```

1  hits  $\leftarrow$  0
2  misses  $\leftarrow$  0
3  equals  $\leftarrow$  0
4  for all edges in  $G$ , whose source vertex (given as  $a_s$ ) and target vertex
   (given as  $a_t$ ) are not the same
5      do  $\gamma \leftarrow t(a_s, a_t)$ 
6          remove the edge  $(a_s, a_t)$ 
7          find an alternate path,  $p(a_s, a_t)$  from  $a_s$  to  $a_t$ , such that the length of
            $p(a_s, a_t)$  is equal to 2, that is,  $p(a_s, a_t) = \langle a_s, a_r, a_t \rangle$  where
            $a_r$  is a recommender vertex, and  $a_s$  and  $a_r$  have a minimum
           of  $z$  outgoing edges
8          if  $p(a_s, a_t)$  exists
9              then  $\alpha \leftarrow t(a_r, a_t)$ 
10                  $\beta \leftarrow \text{trustvalue}(\text{percentile}(t(a_r, a_t), d_{a_r}), d_{a_s})$ 
11                 if  $\alpha = \beta$  or  $|\alpha - \beta| < 0.05$ 
12                     then equals ++
13                 elseif  $|\beta - \gamma| < |\alpha - \gamma|$ 
14                     then hits ++
15                 elseif  $|\alpha - \gamma| < |\beta - \gamma|$ 
16                     then misses ++
17                 restore the edge  $(a_s, a_t)$ 
18  print hits, misses, equals

```

**Fig. 1.** Experiment design

## 5.2 Data Set

We generate a simulated web of trust based on a random graph [30] as described in Figure 2.  $n$  is the number of vertices in the graph,  $k$  is the number of outgoing edges of each vertex, and  $G$  is the generated graph.

As we discussed in Section 2, different source agents may assign different trust values to a target agent. This occurs due to their different dispositions to trust even though their individual experiences with the target agent are the same.

These ideas are reflected in the generation of this web of trust. The trustworthiness value  $q_{u_i}$  represents the experience that other agents would have with agent  $u_i$ . Since  $q_{u_i}$  remains constant for agent  $u_i$ , any agent that interacts with it has the same experience. Although this would not always be true in a real web of trust, placing this condition sets up a suitable controlled environment for our experiments. If there is an instance where the subjectivity-eliminated trust recommendation method is ineffective, we know that it is not because multiple agents may have assigned  $u_i$  different trust values due to different experiences, in which case subjectivity is irrelevant. The failure is in fact on part of the method.

GENERATE-WEB-OF-TRUST( $n, k$ )

```

1  create an empty weighted directed graph,  $G(V, E)$ , where  $V$  is the set of
    vertices and  $E$  is the set of edges
2  populate  $V$  with  $n$  vertices, labeled  $u_i$ , where  $i = 0, 1, \dots, n - 1$ 
3  with each vertex  $u_i$ , associate a random trustworthiness value  $q_{u_i}$ 
    from the interval  $[0,1]$ 
4  with each vertex  $u_i$ , associate a random skew factor  $s_{u_i}$  from the interval  $[0,2]$ 
5  for each vertex  $u_i$ 
6      do select  $k$  random distinct vertices from  $V$ , refer to them as  $v_j$ , where
         $j = 0, 1, \dots, k - 1, u_i \neq v_j$ 
7      for each vertex  $v_j$ 
8          do create the edge  $(u_i, v_j)$  in  $E$ 
9          assign the weight  $power(q_{v_j}, s_{u_i})$  to  $(u_i, v_j)$ 
10 return  $G$ 

```

**Fig. 2.** Pseudo code for generating the web of trust

The skew factor represents the individual disposition to trust of each agent. Although different agents have the same experience with a given agent  $u_i$ , they each assign it a different trust value based on their own disposition to trust. If the skew factor  $s_{u_i}$  is less than 1,  $q_{v_j}$  would be skewed upwards. Otherwise if the skew factor  $s_{u_i}$  is greater than 1,  $q_{v_j}$  would be skewed downwards.

Weights or trust values are drawn from the set of real numbers between 0 and 1 therefore the resolution for expressing trust is high.

The resulting data set is a web of trust where we know that subjectivity in fact does exist.

The web of trust consists of  $n$  vertices and  $n \cdot k$  edges. If the number of vertices is 1000 and  $k = 100$ , the total number of edges is  $n \cdot k = 100,000$ . A new web of trust is generated for each run according to the values of  $n$  and  $k$  under consideration. The number of outgoing edges for all vertices is exactly  $k$ , therefore  $z = k$ .

### 5.3 Experiment Runs and Observations

The results of two sets of experiment runs are given in Table 1 and Table 2. We note that with  $n = 1000$ , and  $z = k = 180$ , 95% of the time, the subjectivity-eliminated trust recommendation method gives better results than those given by the unmodified trust recommendation method (not considering instances when both methods give equal results).

We also note that increasing  $z$  improves the effectiveness of the method. However, increasing  $n$  while keeping  $z$  constant (that is, decreasing the connectivity of the graph) does not seem to deteriorate the effectiveness of the method.



**Table 1.** Experiment runs with  $n = 1000$ 

$z, k$	<i>hits</i>	<i>misses</i>	<i>equals</i>	$\frac{\textit{hits}}{\textit{hits}+\textit{misses}}$
10	0	0	0	-
20	0	0	0	-
30	16345	3568	6376	82%
40	39246	7371	15531	84%
50	80191	12439	29936	87%
60	141860	20251	50283	88%
70	223511	29094	85819	88%
80	332837	43046	130526	89%
90	488874	52617	180220	90%
100	674139	63542	253553	91%
110	903407	85568	331536	91%
120	1175525	97396	441145	92%
130	1520318	107460	554661	93%
140	1892642	137848	698261	93%
150	2383352	142981	830549	94%
160	2809821	181346	1084773	94%
170	3450976	195444	1242734	95%
180	4154572	203933	1448044	95%

**Table 2.** Experiment runs with  $z = k = 100$ 

$n$	<i>hits</i>	<i>misses</i>	<i>equals</i>	$\frac{\textit{hits}}{\textit{hits}+\textit{misses}}$
1000	674139	63542	253553	91%
1200	673636	65947	251049	91%
1400	683320	64536	241659	91%
1600	680652	66192	246285	91%
1800	682642	64880	243262	91%

## 6 Discussion of Experiment Results / Future Work

The results of the experiment runs on the simulated web of trust provide a positive indication that the subjectivity-eliminated trust recommendation method is more effective than the unmodified method. Our method gives significantly better results when the number of outgoing edges of the agents is high. Even with relatively lower number of outgoing edges, the method still outperforms the one that does not account for subjectivity.

However, despite the strength of the results we can only consider them as a positive initial indication of the effectiveness of the method. Concrete conclusions are not feasible at this stage due to the reason that the simulated web of trust is a simplistic approximation of a real web of trust. Although the simulated web of trust takes into account real world issues such as rater bias, some other important aspects are simplified. For example, it is wired simply as a

random graph. Another simplification is the uniform distribution of the dispositions to trust and the trustworthiness of the agents. These simplifications may very well not have any impact on the effectiveness of the method however that is a hypothesis which needs to be tested.

An evident direction for future work is to test the method on a real web of trust or a closely approximated simulated web of trust. Some ideas for generating a more realistic web of trust include: 1) connectivity based on small-world [31] or scale-free networks [32], which are better representations of social networks, and 2) representing the dispositions to trust and the trustworthiness of the agents by a distribution such as normal or power-law. These ideas were not incorporated into this paper since their implementation is not straightforward. Most of the work on small-world and scale-free networks relates to undirected graphs. Methods for generating directed graphs are often for citation-like networks where older nodes do not have edges to newer nodes, which is not the case in a web of trust. So far we have also not come across any existing studies on the distributions of disposition to trust and trustworthiness.

## 7 Conclusion

This paper delved into the problem of subjectivity in trust recommendation, which we argued prevents the real meaning of a trust value from being conveyed by one agent to another. We presented a method for the elimination of subjectivity from trust recommendation that takes advantage of trust scores given as percentiles, which are equally meaningful among two agents. Experiments conducted on a simulated web of trust demonstrated that the method is highly effective for elimination of subjectivity from trust recommendation. The method is non-intrusive and does not require any change in how agents locally evaluate other agents. Furthermore, the method does not involve any third party mediation, thus making it suitable for decentralized networks. Validation of the experiment results on a real web of trust or a closely approximated simulated web of trust is proposed as future work. It is our hope that this paper will also serve as an introduction to the problem of subjectivity in trust recommendation and that it will inspire further research on this problem which has not received considerable attention.

## References

1. Hoffman, K., Zage, D., Nita-Rotaru, C.: A survey of attack and defense techniques for reputation systems. Technical Report CSD TR 07-013, Department of Computer Science, Purdue University, IN, USA (2007)
2. Abdul-Rahman, A., Hailes, S.: Supporting trust in virtual communities. In: Proceedings of the 33rd Hawaii International Conference on System Sciences (January 2000)
3. Baras, J.S., Jiang, T.: Cooperation, Trust and Games in Wireless Networks. In: Advances in Control, Communication Networks, and Transportation Systems: In Honor of Pravin Varaiya. Birkhauser, Boston (2005)

4. Beth, T., Borcherding, M., Klein, B.: Valuation of trust in open networks. In: Gollmann, D. (ed.) ESORICS 1994. LNCS, vol. 875. Springer, Heidelberg (1994)
5. Capra, L.: Engineering human trust in mobile system collaborations. In: Proceedings of the 12th ACM SIGSOFT International Symposium on Foundations of Software Engineering, Newport Beach, CA, USA (2004)
6. Griffiths, N.: Task delegation using experience based multidimensional trust. In: Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems, The Netherlands (2005)
7. Meng, K., Zhang, X., chun Xiao, X., du Zhang, G.: A bi-rating based personalized trust management model for virtual communities. In: Proceedings of the 2006 IEEE International Conference on Networking, Sensing and Control (ICNSC 2006) (2006)
8. Marsh, S.P.: Formalising Trust as a Computational Concept. PhD thesis, Department of Mathematics and Computer Science, University of Stirling, Scotland, UK (1994)
9. Golbeck, J., Hendler, J.: Filmtrust: Movie recommendations using trust in web-based social networks. In: Proceedings of the 3rd IEEE Consumer Communications and Networking Conference (CCNC 2006), Las Vegas, NV, USA (2006)
10. Gambetta, D.: Can We Trust Trust? In: Trust: Making and Breaking Cooperative Relations, pp. 213–237. Department of Sociology, University of Oxford (2000)
11. Toivonen, S., Lenzi, G., Uusitalo, I.: Context-aware trust evaluation functions for dynamic reconfigurable systems. In: Proceedings of the WWW'06 Workshop on Models of Trust for the Web (MTW 2006) (May 2006)
12. Jonker, C.M., Treur, J.: Formal analysis of models for the dynamics of trust based on experiences. In: Garijo, F.J., Boman, M. (eds.) MAAMAW 1999. LNCS, vol. 1647. Springer, Heidelberg (1999)
13. Levien, R.: Attack resistant trust metrics. Manuscript, University of California - Berkeley (2002), <http://www.levien.com/thesis/compact.pdf>
14. Resnick, P., Zeckhauser, R.: Trust among strangers in internet transactions: Empirical analysis of ebay's reputation system. *The Economics of the Internet and E-Commerce* 11 (2002)
15. McKnight, D.H., Choudhury, V., Kacmar, C.: Developing and validating trust measures for e-commerce: An integrative typology. *Information Systems Research* 13(3), 334 (2002)
16. Rotter, J.B.: A new scale for the measurement of interpersonal trust. *Journal of Personality* 35(4), 651 (1967)
17. Rotter, J.B.: Generalized expectancies for interpersonal trust. *American Psychologist* 26(5), 443 (1971)
18. Boon, S.D., Holmes, J.G.: The Dynamics of Interpersonal Trust: Resolving Uncertainty in the Face of Risk. In: *Cooperation and Prosocial Behaviour*, p. 190. Cambridge University Press, Cambridge (1991)
19. McCord, M., Ratnasingam, P.: The impact of trust on the technology acceptance model in business to consumer e-commerce. In: Proc. of the Intl. Conf. of the Information Resources Mgmt. Association: Innovations Through IT, New Orleans, LA, USA (May 2004)
20. Kaluscha, E.A.: The Importance of Initial Consumer Trust in B2C Electronic Commerce - A Structural Equation Modeling Approach. PhD thesis, University of Klagenfurt, Austria (2004)
21. Ziegler, C.N., Lausen, G.: Spreading activation models for trust propagation. In: Proceedings of the 2004 IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE 2004) (2004)

22. Golbeck, J., Parsia, B., Hendler, J.: Trust networks on the semantic web. In: Klusch, M., Omicini, A., Ossowski, S., Laamanen, H. (eds.) CIA 2003. LNCS, vol. 2782, pp. 238–249. Springer, Heidelberg (2003)
23. Chen, R., Yeager, W.: Poblano: A distributed trust model for peer-to-peer networks. Technical report, Sun Microsystems (2000)
24. Abdul-Rahman, A., Hailes, S.: A distributed trust model. In: Proceedings of the 1997 Workshop on New Security Paradigms, Langdale, Cumbria, United Kingdom (1997)
25. Christie, R., Geis, F.L.: Studies in Machiavellianism. Academic Press, New York (1970)
26. Hirsh, H., Basu, C., Davison, B.D.: Learning to personalize. Communications of the ACM 43(8), 102 (2000)
27. Langley, P.: Adaptive user interfaces and personalization (2008), <http://www.isle.org/~langley/adapt.html> (retrieved January 5, 2009)
28. Fog, A.: Branch prediction in the pentium family (2008), <http://x86.org/articles/branch/branchprediction.htm> (retrieved January 5, 2009)
29. National Institute of Standards and Technology (NIST): Nist/sematech e-handbook of statistical methods - percentiles (2008), <http://www.itl.nist.gov/div898/handbook/prc/section2/prc252.htm> (retrieved January 5, 2009)
30. Bollobas, B.: Random Graphs. Cambridge University Press, Cambridge (2001)
31. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. Nature 393 (June 1998)
32. Albert, R., Barabasi, A.L.: Statistical mechanics of complex networks. Reviews of Modern Physics 74 (January 2002)

# Security in Wiki-Style Authoring Systems

Christian Damsgaard Jensen

Department of Informatics and Mathematical Modelling

Technical University of Denmark

Christian.Jensen@imm.dtu.dk

**Abstract.** During the past decade, online collaboration has grown from a practice primarily associated with the workplace to a social phenomenon, where ordinary people share information about their life, hobbies, interests, politics etc. In particular, social software, such as open collaborative authoring systems like wikis, has become increasingly popular. This is probably best illustrated through the immense popularity of the Wikipedia, which is a free encyclopedia collaboratively edited by thousands of Internet users with a minimum of administration.

As more and more people come to rely on the information stored in open collaborative authoring systems, security is becoming an important concern for such systems. Inaccuracies in the Wikipedia have been rumoured to cause students to fail courses, innocent people have been associated with the murder of John F. Kennedy, etc. Improving the correctness, completeness and integrity of information in collaboratively authored documents is therefore of vital importance to the continued success of such systems.

It has previously been observed that integrity is the most important security property in open collaborative authoring systems. In this paper we propose a general security model for open collaborative authoring systems based on a combination of classic integrity mechanisms from computer security and reputation systems. The model is able to accommodate a number of different integrity policies and three different policies are presented in the paper. While the model provides a reputation based assessment of the trustworthiness of the information contained in a document, the primary objective is to prevent untrustworthy authors from compromising the integrity of the document. In order to determine the effectiveness of the proposed integrity model, we present an attacker model for open collaborative authoring systems, which allows us to calculate the vulnerability of a given document based on the fraction of malicious authors in the system.

## 1 Introduction

Collaborative authoring systems which support an open and dynamic population of authors, such as the Wiki [1], have become increasingly popular over the past couple of years. Large pieces of documentation, such as the Wikipedia [2], have been compiled using this type of technology and the Wiki technology has become an indispensable part of many computer supported collaborative work (CSCW) tools that support a distributed user base. While it may be argued that collaboratively authored documents will never have the same authority as a traditionally edited work [3,4], the Wikipedia project has demonstrated the benefits of this approach by compiling a comprehensive

and largely accurate encyclopedia from the contributions of individual people located around the world. However, the Wikipedia has also exposed one of the weaknesses of collaborative authoring, which is that malicious or incompetent users may compromise the integrity of the document by introducing erroneous entries or corrupting existing entries, e.g., Jimmy Wales, the co-founder of the Wikipedia, claims to receive 10 emails every day from students who failed their courses because the information cited from the Wikipedia turned out to be wrong [5] and public figures sometimes find that the entry describing them in the Wikipedia has been modified to defame them [6,7]. Despite the contested findings in Nature [8], which found that the quality of information in the Wikipedia was almost as high as the Encyclopaedia Britannica<sup>1</sup> it appears obvious that a general mechanism to improve the quality of documents produced in open collaborative authoring systems is needed. The quality of a collaboratively authored document is determined by a few simple properties, i.e., is the document complete, correct and unbiased. We have previously argued that these properties correspond to the properties ensured by existing integrity mechanisms in computer security [10], so we intend to leverage this work by designing an integrity mechanism for open collaborative authoring systems. Most data protected by an integrity mechanism, however, have well defined syntax and semantics, whereas the syntax and semantics of collaboratively authored documents are difficult to define. This means that existing integrity mechanisms cannot be used directly. The obvious answer to this problem is to rely on feedback from the users, i.e., some reputation system similar to the ones used by Amazon [11], eBay [12] or the “WOT” plugin for Firefox [13]. Relying on external feedback corresponds to the approach, which is already used in a wiki (cf. Section 2.2), where other authors may revert a document to an earlier version. Reputation systems<sup>2</sup> have previously been proposed as an effective means to assess the quality of information from uncertain sources [15,16,17,18,19], but they only help automate detection of undesirable content and are generally unable to prevent undesirable content from being introduced into the document. We therefore propose a combination of reputation systems to assess the quality of collaboratively authored documents and traditional integrity mechanisms to prevent unknown or untrusted users from modifying the documents in the collaborative authoring system. The mechanism automatically assigns a “quality rating” to every document in the system, based on the reputation of the last user who updated the document. In order to enforce integrity, we want that only users with a similar or higher reputation than the past user will be able to modify the entry. This means that users with a poor reputation will be unable to update most of the documents in the systems, but more importantly that documents that have a high quality rating may only be modified by the few users who have an equally high reputation.

The structure of the rest of the paper is as follows. We start, in Section 2, with a definition of our model of an open collaborative authoring system and identify important properties of such systems. Section 3 presents a short overview of the most important integrity mechanisms developed in the context of computer security. We propose our integrity model for open collaborative authoring systems in Section 4. This model is

---

<sup>1</sup> The Encyclopaedia Britannica issued a 20 page response [9] to the article in Nature, which questions the methods used in the study published in Nature.

<sup>2</sup> A good survey of reputation systems was published by Jøsang, Ismail and Boyd [14].

based on a document review process, which is described in Section 5 and an evaluation of the proposed model is presented in Section 6. Directions for our future work are outlined in Section 7 and our conclusions are presented in Section 8.

## 2 Open Collaborative Authoring Systems

Systems that allow online communities to author and publish collaborative work can be organised in different ways. We present an outline of a generic architecture in the following, which defines our terminology and allows us to identify different properties of such systems.

### 2.1 System Model

An open collaborative authoring system (OCAS) is defined by a backbone server network and a set of client machines that are used by the authors to access the documents. This architecture is illustrated on Figure 1.

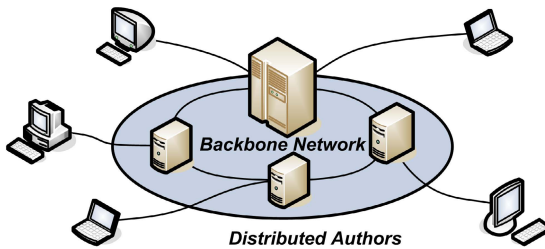


Fig. 1. System Model

The backbone network, which may consist of a single server, stores all data and mediates users' access to documents including enforcement of access control policies and synchronisation when multiple authors simultaneously wish to update the same document. The backbone network infrastructure may either be universally accessible or it may belong to a closed community, such as a company's internal intranet, which limits the set of clients that are able to access the infrastructure.

The client machine runs the necessary software to allow a distributed set of authors to read and write documents stored in the backbone network. In many cases, this software is simply a web-browser, which allows the client to interact with the system that runs on a web-server in the backbone network. Authors may be required to register an identifier that is used to identify the contributions of the different authors in the system.

### 2.2 System Security Model

Open collaborative authoring systems often allow any user who can access the system to create an account (possibly anonymously) and start creating new or editing existing documents. The most popular open authoring technology is probably based on

Wiki technology, so we focus our analysis on policies and mechanisms developed for the wiki.<sup>3</sup> The basic philosophy behind a wiki is that everyone should be allowed to edit everything, but that it should be easy to restore the document to its prior state if the modifications are considered undesirable. The traditional security process is based on *prevention, detection and response*, where security mechanisms are introduced to prevent unauthorised access to protected resources, auditing procedures and intrusion detection systems are introduced to detect unauthorised use of the system and a combination of automatic and manual procedures are used to stop unauthorised access and return the system to a consistent state. Applying this process to the wiki philosophy, we see that there are few mechanisms to prevent malicious or accidental modification of a wiki page; detection is left to the users and the only means of response is to restore the previous version of the page.

Authors in wiki-style systems are often required to create an account before they can edit pages, but this is not an essential requirement. Moreover, authors may create multiple accounts and it is generally not possible to identify the person who registered a particular account. This means that the primary means of authentication is the password that the author provided when he created his account, i.e., all information needed to create an account may have been provided by an otherwise anonymous user. This results in a lack of accountability, which was exploited by unscrupulous users in the examples listed in the Introduction. The only effective means of recognising users in most wikis is therefore the IP-address of the client machine, which allows malicious users operating from a fixed Internet address to be identified and blocked, but users operating from a machine with a dynamically allocated IP-address and those who operate from public access terminals, e.g., public libraries or Internet cafés, cannot be blocked.

The threshold to enter a wiki is often relatively low, e.g., all users are granted the same access privileges, so the cost of discarding a compromised account identifier and create a new account is equally low. This means that wiki-style systems are extremely vulnerable to the Sybil attack [20], where an attacker may create a new identifier for every attack. It is difficult to completely avoid the Sybil attack without logically centralised identity management, e.g., a public key infrastructure (PKI), but it is possible to reduce the problem by increasing the cost of generating new identities. This approach would be similar to HashCash [21] and “proof-of-work” systems [22] used to fight Spam email.

### 3 Integrity Mechanisms in Computer Security

Integrity mechanisms are designed to prevent corruption or destruction of data managed by the system. This means that mechanisms should be in place to prevent malicious users from tampering with data directly and to ensure that data is kept coherent, e.g., all modifications of data should be through well-formed transactions [23].

One of the first models to address the problem of integrity was proposed by Biba, who defined an integrity model [24] analogous to the well known mandatory access

---

<sup>3</sup> Although our definition of an open collaborative authoring system is not limited to existing wiki-style systems, we sometimes use the terms wiki and wiki-style systems in place of the much longer open collaborative authoring systems.



control model defined by Bell & LaPadula [25]. The model divides subjects (processes running on the behalf of named users) and objects (stored data manipulated by the processes) into different integrity classes and defines the following two security properties<sup>4</sup>

1. Simple security property, which states that a subject at a given level of integrity may not read an object at a lower integrity level (**no read down**).
2. \* (star) security property, which states that a subject at a given level of integrity must not write to any object at a higher level of integrity (**no write up**).

The model prevents flow of information from low integrity documents into high integrity documents. The subjects defined by the Biba model correspond to the browsers used to update the wiki and the objects of the Biba model correspond to the documents in the wiki. Most operating systems, in common use on the Internet, do not distinguish between the access rights of different processes started by the same user, e.g., a user may freely *cut-and-paste* between all the GUI-windows on the monitor, so there is no reason to distinguish between a process and the user who launched it, i.e., a subject in the Biba model corresponds to an author in the OCAS.

The Biba model works well when the security mechanism has complete mediation, i.e., every access to every object must be checked for authority. In wiki-style systems, however, authors may have several programs open at the same time, so it will be practically impossible to enforce the simple security property, but we believe that the \*-security property may prove useful.

Another aspect of the Biba integrity model which we believe will prove useful is the concept of a “low-watermark,” which changes the integrity class of the subject to the integrity class of the lowest object accessed by the subject. As we mentioned above, we do not expect complete mediation, so we will not be able to correctly update the integrity level of subjects, instead we propose to interpret the low-watermark policy with respect to the object, which means that we change the integrity class of the document to the integrity class of the subject with the highest integrity class who accessed the object.

Another influential integrity model was defined by Clark and Wilson who realised that integrity is more important than confidentiality in many commercial and government applications. The Clark-Wilson integrity model [23] is not explicitly based on integrity levels, but is instead based on the notion of well-formed transactions that transform a system from one consistent state to another. This means that integrity is not considered an inherent property that can be used to label subjects and objects. Instead, integrity is defined by the relationship between certain object and it can be enforced by imposing constraints on the operations that subjects can perform on objects. This model is not directly applicable in collaborative authoring systems, because it appears difficult to define criteria for consistency of text documents and few general text editing methods can be considered well-formed transactions.

The low-watermark integrity policy has recently been explored in the context of commercial off-the shelf software [26]. The LOMAC integrity mechanism is implemented

---

<sup>4</sup> These security properties are the inverse of the properties defined in the Bell & LaPadula model, which focus on confidentiality instead of integrity.

as a Linux kernel extension, which supports low water-mark integrity policies in a standard Unix system. The default policy only defines two integrity levels for data: high for system files and devices, except the network interface card, and low for user data. Integrity policies with more integrity levels can be defined when the system is installed. Processes inherit the integrity level of the user who started them, but if they access data with a lower integrity level, their own integrity level is demoted to that of the data.

## 4 Security in Open Collaborative Authoring Systems

In order to improve the quality of documents in an open collaborative authoring system, we define an integrity mechanism that limits the set of documents that an author can update. The integrity mechanism is based on two basic integrity models: the *static integrity model* and the *dynamic integrity model*, which capture respectively the static and dynamic properties of integrity control.

### 4.1 Static Integrity Model

The static integrity model defines whether a registered author should be allowed to edit a particular document based on the quality of the author's previous contributions and the estimated quality of the document. As all users should be allowed to read all information in an open collaborative authoring system, the static integrity model defines the actions of the *reference monitor*, i.e., it defines the access control model.

All authors must have an identifier<sup>5</sup> (possibly a pseudonym) which will allow the system to recognise authors and attribute them with a quality confidence value (QCV), which indicates the normal level of correctness, completeness and lack of bias in documents by that author, i.e., it encodes the reputation of that author. Similarly, each section of the document must have an integrity level (IL) associated with it, which indicates the level of correctness, completeness and lack of bias in that particular document. There is an obvious relationship between the QCV of an author and the IL of the (sections of) documents that she has authored<sup>6</sup> because authors of documents with a high IL must have an equally high QCV to ensure that the quality of documents is preserved and documents edited by authors with a high QCV are likely to improve in quality, so the IL of the edited document should be equally high. The IL of a document is determined partly by the QCV of the authors who have contributed to the document and partly by feedback from registered authors in the system. This means that the integrity level (QCV) of the last author to edit a document determines the current integrity level (IL) of that document. The integrity label of the document is modified to reflect the integrity label of the author, which is the opposite of the low water-mark policies described in Section 3. However, we believe that it is reasonable to assume that authors who have a history of writing complete, correct and unbiased documents are likely to

<sup>5</sup> We do not require that authors have unique identifiers, so an author may have multiple identifiers but we do expect that authors do not share identifiers.

<sup>6</sup> Without loss of generality we consider documents with a single author in the following, but the integrity model is trivially extended to smaller textual units, such as sections or paragraphs, so that documents with multiple authors may be encompassed by the model.

continue in that style, so new documents edited by such authors will benefit from their involvement, i.e., the document will be raised to the high level of the author.<sup>7</sup> The user feedback mechanism allows documents to be promoted beyond the level of the authors who have contributed to the document as defined by the dynamic integrity model described below.<sup>8</sup>

Formally, the static integrity model defines the concepts of *authors*, *documents*, *quality confidence values* and *integrity levels* as:

$A$  is the set of identifiers of authors who have registered to use the system.

$D$  is the set of documents that are managed by the system.

$I$  is a totally ordered set of integrity levels, such as {low, medium, high} or  $[0, 4]$ , with the ordering relation “ $\leq$ ” defined in the usual way, e.g., “low  $\leq$  medium  $\leq$  high” and “ $0 \leq 1 \leq 2 \leq 3 \leq 4$ ”.<sup>9</sup>

We also define two functions  $qcv(a : A) : A \rightarrow I$  and  $il(d : D) : D \rightarrow I$  which allow us to compare the QCV of an author directly with the integrity level of a document using the total order on  $I$ :

$$qcv(a : A) = \{\text{quality confidence value of } a\},$$

$$il(d : D) = \{\text{integrity level of } d\}.$$

Finally, we define the predicate:

$$can\_edit(a : A, d : D) = \{ '1' \text{ iff } il(d) \leq qcv(a) \},$$

which returns ‘1’<sup>10</sup> if the author is allowed to edit the document.

Intuitively, an author should only be allowed to edit a document if her reputation is higher than the current quality of the document, so authors should only be allowed to modify documents when the *can\_edit* predicate is ‘1’. This prevents information generated by “low integrity” authors from entering “high integrity” documents, which corresponds to the \*-property in the Biba model. Together with a low water-mark policy defined by the dynamic integrity model below, the \*-property ensures that the integrity label of a document can only increase.

Finally, we do not associate a particular semantics with the different integrity labels and we do not prescribe a pre-defined number of labels in the model.

<sup>7</sup> In order to simplify our presentation, we assume that all documents in the wiki relate to the same subject area, so an author is equally competent to edit all documents. The model is easily extended to encompass documents with different subjects by introducing a classification of documents according to subject and maintaining separate QCV for each subject that an author contributes to.

<sup>8</sup> The user feedback mechanism should also allow the IL of a document to be lowered, if it turns out that the trust in the author was misplaced.

<sup>9</sup> The ordering relation “ $\leq$ ” is similar to the *dominance relations* defined by the Bell & LaPadula and Biba models.

<sup>10</sup> We generally use the normal programmer’s convention of representing the boolean values ‘true’ as ‘1’ and ‘false’ as ‘0’.

## 4.2 Dynamic Integrity Model

The dynamic integrity model controls the modification of quality confidence values and integrity levels based on the *integrity watermark model* outlined above and a *document review model*, which controls the explicit promotion and demotion of documents based on feedback from the other authors in the system. These models are described in greater details in the following.

### Integrity Watermark Model

As mentioned above, authors with a high QCV are likely to improve the quality of documents with a lower IL, so we define a model, based on the classic watermark model, to increase the IL of documents that have been modified by good authors and increase or decrease the QCV of authors who have contributed to documents that change their IL as part of a document review (cf. Section 5).

We define the function  $\mathcal{E}(a : A, d : D) : A \times D \rightarrow D$ , which is invoked when author  $a$  wish to edit document  $d$ . The changes to the IL of  $d$  are not represented in the signature for  $\mathcal{E}$ , because they are really side effects of editing the document. However, since the focus of this paper is on security, we only considered these aspects in our definition of  $\mathcal{E}$ .

$$\mathcal{E}(a : A, d : D) = \begin{cases} il(d) := qcv(a), & \text{if } can\_edit(a, d), \\ security\ violation, & \text{otherwise.} \end{cases}$$

The integrity watermark model allows good authors to raise documents to their own integrity level, but it has no effect on the QCV of the other authors who contributed to the document. We therefore introduce a mechanism that allows an author to improve her QCV by submitting a number of documents, which she has contributed to, for a *document review* by other authors (some of these authors *must* have a higher QCV in order to protect against Sybil attacks.) This review mechanism implements the document review model defined below.

### Document Review Model

Any author who has contributed to a document can request a document review, which will determine whether the IL of the document should be increased as a result of the modifications that have been made since the last document review; this is known as a *promotion* of that document. If the documents are promoted, the principal author of the documents will also be promoted. The *principal author* may be defined in different ways, e.g., it may be the author who contributed most modifications since the last promotion, the author who contributed the latest modifications, the contributing author who proposed the promotion of the document or any combinations of the above. It is also possible that modifications reduce the quality of a document, so it must be possible to decrease the IL of the document; this is known as a *demotion* of that document. All users who can edit a document, can also request a document review, which will determine whether the document should be demoted. This means that all authors who *can\_edit* a document may request a demotion review, while only authors who have actually contributed to a document may request its promotion. We believe that this

strikes a reasonable balance between integrity of the document and the threat of denial of service through spurious document review requests. If a document is demoted, the QCV of the principal author will be reduced accordingly. In this paper, we propose to promote/demote the author who contributed most to the document, but this may not be appropriate for all collaboratively authored documents, so we aim to study the effects of different promotion-/demotion strategies on the dynamics of collaboration in future work.

As already mentioned, wiki-style systems are vulnerable to Sybil attacks, where an attacker registers an author identifier, which is then used to corrupt data in the system. When the culprit is found and the identifier has been banned from the system, the attacker registers a new identifier and continues to corrupt the system. Moreover, the attacker may simultaneously register multiple identifiers and use this to orchestrate collusion among seemingly unrelated authors. It is therefore important that new users are introduced at the lowest integrity level, so that they are unable to corrupt documents that have already been reviewed or improved by good authors. It is equally important to ensure that an attacker who registers many new author identifiers is unable to improve the QCV of any of these identifiers through the document review mechanism, so we require that authors with a higher QCV is involved in the review of each document. Moreover, the work required to raise a sufficient number of author identifiers to a QCV level that would allow an attacker to control promotions from lower levels must therefore be high (cf. Section 6.2).

The set of integrity levels ( $I$ ) with its total ordering, defines a natural hierarchy among authors and documents. As mentioned above, new authors must be introduced at the lowest integrity level and are initially only able to create and contribute to documents as this level. In order to be promoted, they have to submit high quality documents for a document review. By requiring that authors with a higher QCV participate in the review, we help protect against the Sybil attack described above and each level in the hierarchy that is involved in the review increases the robustness against this form of attack. The number of levels in the hierarchy depend on the number of elements in  $I$  and the number of levels that must be involved depend on the desired balance between document integrity and the workload needed to review documents. As there is no protection against malicious authors at the initial level, we believe that the number of levels involved should be at least 3. Moreover, we observe that the top levels in the hierarchy cannot be protected by the levels above, so the distance from top to bottom must be sufficiently large to increase the difficulty for an attacker to control the highest levels by getting a sufficient number of malicious authors promoted. This is the reason that we propose at least 5 levels of authors, so that there must be at least 5 different integrity levels in  $I$ . In the following we describe a document review system with exactly 5 levels, numbered from '0' to '4'.

## 5 Document Review

The following document review process is defined to protect the integrity of documents in an OCAS. The documents are divided into several integrity classes according to the

integrity models defined above and each level  $L_i$  of the system is characterised by an index  $i$ . We assume that each level in the hierarchy contains  $\Lambda_i$  registered authors and we assume that  $z_i$  of these authors are malicious and in collusion with each other. For the purpose of our discussion, we assume that only malicious authors will act improperly during the reviewing session of a document. During a document review,  $r_i$  of the  $\Lambda_i$  authors are randomly selected (without any bias) in order to review the document submitted by a user of the same or a lower level. Each of these reviewers determine whether they believe that the document should be promoted/demoted and return their verdict to the document review system. A certain majority of reviewers at a given level  $L_i$  must approve the quality of the submitted document, so that it will be accepted by all level  $\Lambda_i$  users. We can call this majority  $\tau_i$ . Here, we can distinguish between a simple system, and a weighted one. In the weighted system, the vote of an author at level  $L_i$  may have a positive weight of  $w_i$  depending on her past performance. In the following, we limit our analysis to the simple review system, where all authors votes are weighted equally.

We define a set of numerical integrity levels where the lowest level is 0 ( $L_0$ ) and that the highest level is 4 ( $L_4$ ) with the total order defined by ' $\leq$ ',<sup>11</sup>

We consider  $\partial_j(d)$  to be the judgement of reviewer  $j$  for the document  $d$ . Judgements can take two values: rejection or approval, '0' or '1'. In order to reason about the quality of judgements, we assume that the quality of a document can either be poor (in this case the article should be rejected and we write " $d = 0$ "), or high (in this case the article should be accepted and we write " $d = 1$ "). There are thus two different types of mistakes that a reviewer can make: A reviewer can either approve an article with low quality, or reject an article with high quality. These two errors can be expressed as  $\partial_j(d = 0) = 1$  and  $\partial_j(d = 1) = 0$ , respectively.

We continue our study by defining the overall decision of the document review system upon a reviewed document. When a document at one of the three lower levels (0, 1, 2) is reviewed, the reviewers are selected from the same level as the document and the two levels above. When an article of level 3 is reviewed, reviewers are selected from the same and the higher level (level 3 and level 4). When an article of level 4 is reviewed, then all reviewers are selected from the same level.

Let  $\mathcal{D}_i(d)$  be the combined judgement of the reviewers at level  $L_i$  about a document  $d$ . Then  $\mathcal{D}_i(d)$  can take two values: rejection or approval, which we write '0' or '1' respectively. The judgement at a specific level depends on the judgement of the subset  $\Lambda_{R_i} \subseteq \Lambda_i$ , that includes  $r_i$  randomly selected reviewers from  $\Lambda_i$ , as:

$$\mathcal{D}_i(d) = \begin{cases} 1, & \text{if } \sum_{j \in \Lambda_{R_i}} \partial_j(d) \geq \tau_i \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

We now define  $D(d)$  as the final decision (judgement) of the reviewing session of the system about a document  $d$ ;  $D(d)$  takes two values: rejection or approval, i.e., '0' or '1'. We are particularly interested in documents submitted by users of the three lowest levels (i.e.,  $L_0$ ,  $L_1$ , and  $L_2$ .) because we are able to define many different policies governing the final decision  $D(d)$  depending on the outcome of the judgement of the reviewers at

<sup>11</sup> There is a simple mapping from any totally ordered set of 5 integrity levels to the set defined above.

each level  $L_i : D_i(d)$ . The definition and analysis of these policies are, however, beyond the scope of this paper, so we limit our analysis to a single simple policy, which we will call  $II_1$ . This policy requires the acceptance of the document by a simple majority of the levels involved in the review, which means approval by any two of the three sets of reviewers from levels  $L_i, L_{i+1}, L_{i+2}$ , i.e., we require that at least two of the  $D_i(d), D_{i+1}(d), D_{i+2}(d)$  have the value '1'. We express this policy in Equation 2.

$$D(d) = (D_{i+2}(d) \wedge D_{i+1}(d)) \vee (D_{i+1}(d) \wedge D_i(d)) \vee (D_i(d) \wedge D_{i+2}(d)) \quad (2)$$

When it comes to documents submitted by authors at the two highest levels (that is  $L_3$  and  $L_4$ .) we have to follow a different approach for two reason. First of all, there are not “many” levels that are higher, as mentioned above and secondly, these levels *must* be guaranteed to contain high quality documents and have honest reviewers, because they guarantee all the lower levels in the system.

Continuing our analysis, when a document at  $L_3$  is reviewed, we define a simple policy for the outcome of the review process ( $D(d)$ ), which requires the acceptance of the document by both sets of reviewers from levels  $L_3$  and  $L_4$ . Thus, we require that  $D_3(d)$  and  $D_4(d)$  have the value '1'. We express this policy as:  $D(d) = D_3(d) \wedge D_4(d)$ . Finally, when a document at  $L_4$  is reviewed, then we require its the acceptance by reviewers from  $L_4$  only, so we have:  $D(d) = D_4(d)$ .

## 5.1 Operation Considerations

The document review system relies on voting, which means that the system must be populated in order to work. In the following, we examine how a system working according to our model may be bootstrapped and how more authors may be added to the system.

All wiki-style systems have an initial author, or an initial group of authors, who decide to decide to start and run the system. Part of the system configuration consists of creating an author identifier for the initial user at each level from  $L_0$  to  $L_4$  – if a group of authors decide to start the wiki, they may distribute these author identifiers among themselves. Having at least one author defined at each level ensures that it is possible to establish the majority of authors at each level that is required by the document review process. It is important that this initial set of authors are never promoted, so we avoid the risk of empty levels in the hierarchy.

When a new author registers in the system, then he is a level  $L_0$  user. After significant contributions, the authors should be promoted to the next level, e.g., from  $L_0$  to  $L_1$ . The required contribution is a number of constructive articles. The term constructive articles refers to high quality articles that are accepted in the system after a document review.

Let  $\alpha_i$  denote the effort (in terms of successfully promoted documents) needed for a single author to be promoted from level  $L_{i-1}$  to level  $L_i$ , where level  $L_0$  represents new authors who have not yet contributed to the system. It must be easy for new authors to start contributing to the system, so we simply require them to register, i.e.,  $\alpha_0 = 0$ . The definition of the effort required for the other values of  $\alpha_i$  ( $0 < i \leq 4$ ) is an important parameter in the security of the system, which we will examine shortly in the evaluation. Furthermore, an author would be demoted after a number of destructive entries ( $\beta_i$ ). The

term destructive articles refers to low quality articles that are demoted in the system after a reviewing session. The index  $i$  in  $\beta_i$  means that an author is demoted from level  $L_i$  to level  $L_{i-1}$ . The definition of the disruption required to demote an author ( $\beta_i$ ) is another important security parameter. We believe that the definitions of both  $\alpha_i$  and  $\beta_i$  are likely to depend on the OCAS, both the integrity requirements for the documents and the size of the system. Moreover, it is possible that these should be dynamic parameters, so that it becomes more difficult to get a document promoted ( $\alpha_i$  increases) if there have been relatively many unsuccessful promotion attempts within a short period of time (this may be an indication that someone is attacking the system). A full analysis of  $\alpha_i$  and  $\beta_i$  is, however, beyond the scope of this paper and we leave this for future work.

## 6 Evaluation

In the following we examine the system's ability to resist attacks from malicious users. We assume that the necessary security mechanisms, such as authentication of registered users and the access control mechanism that implements the static integrity model have been correctly implemented, so we focus on the possible damages caused by registered, authenticated and authorised users.

### 6.1 Attack Model

The reviewing procedure for an article is the following: Every time an article  $d$  is to be reviewed, then  $r_i$  members of a level  $L_i$  are selected as reviewers, in a random manner. A level has  $\Lambda_i$  users, and  $z_i$  of them are malicious. Eventually, the probability that the reviewing process will accept a poor quality article is equivalent to the probability that there are enough malicious reviewers among the  $r_i$  randomly selected ones, so as to bias improperly the outcome of the reviewing process. Let this probability be  $p_i$ ; we can estimate it using Equation 1 and the hypergeometric discrete probability distribution. According to the policy  $\Pi_1$  defined in Equation 2, the simple majority of reviewers at each level who must approve the submitted document ( $\tau_i$ ) is actually a "threshold" for the malicious reviewers in the system. In our study, we are interested in scenarios with at least  $\tau_i$  selected malicious users, because that is the only way colluding malicious authors may control the document review. The number of malicious authors who may participate in a document review is only limited by the number of selected users  $r_i$  and by the total number of malicious users  $z_i$ . We are therefore able to estimate the probability ( $p_i$ ) that a set of  $z_i$  malicious authors may influence the decision of the document review process as:

$$\begin{aligned}
 p_i &= \text{Prob}\{D_i(d) = 1 | d = 0\} \\
 &= \text{Prob}\left\{ \sum_{j \in \Lambda_{R_i}} \partial_j(d) \geq \tau_i | d = 0 \right\} \\
 &= \text{Prob}\{\tau_i \text{ or more of the } r_i \text{ randomly selected reviewers are malicious} | d = 0\} \\
 &= \sum_{k=\tau_i}^{\min(r_i, z_i)} \frac{\binom{z_i}{k} \cdot \binom{\Lambda_i - z_i}{r_i - k}}{\binom{\Lambda_i}{r_i}}
 \end{aligned}$$



The hypergeometric probability distribution is useful when we have to estimate the probability of having exactly  $k$  malicious users among the  $r_i$  that are randomly selected, provided that level  $L_i$  has  $z_i$  malicious users.

In order to illustrate the effect of  $p_i$  in a real system, we consider a small wiki with 32 registered authors at each of the levels. In order for an attacker to influence the decision of the review process, he must control a certain number of authors. For the sake of this experiment we consider the total number of registered authors to be fixed; this is not unrealistic because we could implement a collusion detection system that raises an alarm if many new users are registered within a short period of time. We have analysed three different scenarios, which allows us to examine our policy  $\Pi_1$  for three different sets of parameters. In the first scenario (Scenario 1),  $\Pi_1$  requires that half of the authors at a given level participate in the review (i.e.,  $r_i = 16$ ) and we require a simple majority among the participating reviewers (i.e.,  $\tau_i = 8$ ). In the second scenario, we only require a quarter of the registered authors at a given level to participate and we still require a simple majority among the participating voters (i.e.,  $r_i = 8$  and  $\tau_i = 4$ ). In the third scenario, we again require a quarter of the authors at a given level to participate, but we now require a three quarter majority among the participating voters to approve the document (i.e.,  $r_i = 8$  and  $\tau_i = 6$ ). We have calculated the number of malicious users  $z_i$  that are required at a single level in order to have a certain probability of influencing the outcome of the document review. The result of these calculations is shown in Table 1.

**Table 1.** Number of malicious users that must be controlled to influence decision

	Probability of influencing decision					
	95%	90%	75%	66%	50%	33%
Scenario 1	20	19	17	16	15	14
Scenario 2	22	20	18	16	14	12
Scenario 3	28	27	25	24	22	20

For each scenario, the table shows the minimum number of authors that must be controlled in order to have a specific probability of influencing the outcome of the document review. The table shows that an attacker must control more than two thirds of a level in order to have a higher than 95% probability of influencing the outcome of the document review *at that level*. It also shows that nearly half the authors must be controlled at a particular level in order to have a better than 50% chance of influencing that level. Finally, it shows that changing the simple majority vote to a qualified majority vote, increases the number of authors that must be controlled dramatically.

## 6.2 Cost of Attack

By analysing the dynamic integrity model and the document promotion scheme presented in Section 5, we can now estimate the effort required to launch an attack against the system. When, for example, an attack is launched at level  $L_0$ , and there are  $z_0$ ,  $z_1$ , and  $z_2$  malicious users in levels  $L_0$ ,  $L_1$ , and  $L_2$ , respectively. If an attacker wishes to

have high probability of success, he must try to have as many malicious authors as possible among the authors of the levels involved in the document review. These are the level that he wishes to attack, and quite often, levels that are higher than this. All these imply that he must “pay” a significant effort in order to promote a sufficient number of malicious authors. For the moment, we estimate the needed effort (cost) to have one malicious user in level  $L_i$  as:

$$\sum_{j=0}^i \alpha_j, \text{ where } \alpha_0 = 0 \tag{3}$$

Together with the success probability presented in Section 6.1, the cost of attacking a specific level allows us to calculate an estimate of the vulnerability of an OCAS according to the voting policy chosen in the document review process.

### 6.3 Cost of Attacking System with Policy $\Pi_1$

We are now ready to calculate the cost of an attack against a particular level of a system that implements the document review policy  $P_{i_1}$ .

We base our analysis on the policy specified above, which states that  $D(d) = (D_{i+2}(d) \wedge D_{i+1}(d)) \vee (D_{i+1}(d) \wedge D_i(d)) \vee (D_i(d) \wedge D_{i+2}(d))$ . Therefore, the cost  $\mathcal{C}_0$  of an attack to Level  $L_0$  will be estimated as:

$$\begin{aligned} \mathcal{C}_0 &= \min\{z_1(\alpha_0 + \alpha_1) + z_2(\alpha_0 + \alpha_1 + \alpha_2), z_0\alpha_0 + z_1(\alpha_0 + \alpha_1), \\ &\quad z_0\alpha_0 + z_2(\alpha_0 + \alpha_1 + \alpha_2)\} \\ &= \min\{z_1\alpha_1 + z_2(\alpha_1 + \alpha_2), z_1\alpha_1, z_2(\alpha_1 + \alpha_2)\} \\ &= \min\{z_1\alpha_1, z_2 \sum_{j=0}^2 \alpha_j\} \end{aligned}$$

In a similar manner we can estimate the cost of an attack to the other 4 levels and a summary of the results is shown in Table 2.

**Table 2.** Cost of an attack according to policy  $P_{i_1}$

Level	Cost of attack
$L_0$	$\mathcal{C}_0 = \min\{z_1\alpha_1, z_2 \sum_{j=0}^2 \alpha_j\}$
$L_1$	$\mathcal{C}_1 = \min\{z_1\alpha_1 + z_2 \sum_{j=0}^2 \alpha_j, z_2 \sum_{j=0}^2 \alpha_j + z_3 \sum_{j=0}^3 \alpha_j, z_3 \sum_{j=0}^3 \alpha_j + z_4 \sum_{j=0}^4 \alpha_j\}$
$L_2$	$\mathcal{C}_2 = \min\{z_2 \sum_{j=0}^2 \alpha_j + z_3 \sum_{j=0}^3 \alpha_j, z_3 \sum_{j=0}^3 \alpha_j + z_4 \sum_{j=0}^4 \alpha_j, z_4 \sum_{j=0}^4 \alpha_j + z_2 \sum_{j=0}^2 \alpha_j\}$
$L_3$	$\mathcal{C}_3 = z_3 \sum_{j=0}^3 \alpha_j + z_4 \sum_{j=0}^4 \alpha_j$
$L_4$	$\mathcal{C}_4 = z_4 \sum_{j=0}^4 \alpha_j$

The table above shows us that it is important to define the cost of progressing to the next level ( $\alpha_i$ ) and the voting policies so that a high number of malicious users

( $z_i$ ) is required to control a specific level ( $L_i$ .) Moreover, the document review process should be defined to ensure values of  $\alpha_i$  and  $z_i$  that increases the cost of controlling the different levels, as seen in Table 2.

## 7 Future Work

The model defined and analysed in this paper makes two simplifying assumptions that would seriously impact it's use in real systems. As already mentioned, these assumptions are easily satisfied by trivial extensions to both the static and dynamic integrity models. We discuss these extensions, before we go on to suggest other directions for future work.

The first assumption is that documents are edited by a single author at the time, i.e., all modifications to a document can be attributed to a single author. This assumption is introduced to simplify the discussion of the relationship between the QCV of authors and the IL of the documents that they modify, but the model is trivially extended to cover smaller textual units, such as sections, paragraphs or even sentences. This will allow multiple authors, possibly with different QCVs, to collaborate on a single document. However, managing documents where different sections have different integrity levels raises a number of practical issues that must be addressed in a real system. First of all, we need to determine how the different integrity levels in such documents can be presented to readers in a simple and intuitive way, e.g., by using different background colours [27]. Secondly, we need to investigate the impact of different textual granularities of the integrity mechanism and determine whether there is an optimal granularity or whether the granularity should be a parameter that is configured when the wiki is installed. Finally, we need to determine the impact of a finer textual granularity on the voting protocol, e.g., will reviewers stay vigilant when they are asked to decide whether the third paragraph in a document merits promotion.

The second simplifying assumption is that all authors are assumed to only contribute to documents within their subjects of expertise. This means that we do not have to consider well intended authors who make good contributions to documents within their area of expertise, but poor contributions to documents in other areas, i.e., we only need to manage one QCV for each author. This is easily solved by incorporating a classification scheme, similar to the *Categories* found at the bottom of every article in the Wikipedia, which allows the system to identify the authors areas of expertise (those where the documents are generally promoted.)

In our presentation of the document review process, we confined our analysis to a single simple policy for combining the judgements from different levels in the hierarchy ( $II_1$ ). Other policies are easily formulated and it would be particularly interesting to explore policies that are substantially different from the simple majority rule, e.g., different (possibly dynamic) qualified majorities at the different levels in the hierarchy. The document review process considers all reviewers with the same weight, but it would be interesting to explore the effects of weighted votes where the votes of different reviewers count differently, e.g., through the use of a reputation system. This will have an impact on security, e.g., the model's ability to resist Sybil attacks, so we need to analyse the impact of introducing weighted votes on the overall security of the

system. Finally, we plan to investigate the impact of run-time adaptability of the different security parameters identified in this paper.

## 8 Conclusions

In this paper we addressed the problem of ensuring the quality of documents in open collaborative authoring systems, such as wikis. Existing solutions primarily focus on assessing the quality of the documents themselves, either through an analysis of the documents or through an evaluation of the trustworthiness of the source of the document. While such techniques are extremely useful, they do little to prevent malicious users from corrupting the documents maintained within a community. The lack of authentication in many wiki-style systems leads to a lack of accountability, which results in lower quality of the documents. Instead of introducing a centralised authentication authority, we propose a novel integrity model where authors earn the right to modify documents by contributing to the system, which raises the cost of performing a Sybil attack.

The proposed integrity model combines existing assessment techniques with integrity control mechanisms from computer security, in order to provide quality information to the reader and prevent untrustworthy users from corrupting high quality documents. Documents are internally labelled with an integrity label, which can also be shown to the reader who will then learn something about the provenance of the document and get an idea about whether the content should be trusted. The model also associates integrity labels with authors, which allows a system to prevent authors who have primarily authored low quality documents from modifying documents with a high quality label. The integrity mechanism is designed to ensure that the editing process does not lower the integrity of documents. The model is quite flexible and allows many different policies to be defined with respect to the set of authors selected, quorum and (qualified) majority needed to promote documents to a higher integrity label or demote documents to lower integrity labels.

We presented an analysis of the proposed model, which estimates the probability of a successful attack against a level in the defined integrity hierarchy, given a specific number of malicious authors controlled by the attacker. We also defined the concept of cost of an attack as the effort required to promote a malicious author to the desired level and showed how this may be used to estimate the overall cost of an attack on a system that implements the model. Finally, we have shown how these estimates may be used to determine some of the important security parameters identified above.

## Acknowledgements

We would like to thank several people for their contributions to this paper. Our two colleagues Nicola Dragoni and Robin Sharp, who provided helpful comments and constructive criticism and in particular Petros Kaklamanis, a former student in our group, who contributed to the formalisation of many of the concepts presented in Section 5 and Section 6 of this paper.

## References

1. What is Wiki, <http://www.wiki.org/wiki.cgi?WhatIsWiki> (visited December 28, 2006)
2. Wikipedia, the free encyclopedia, <http://en.wikipedia.org/wiki/Wiki> (visited December 28, 2006)
3. Denning, P., Horning, J., Parnas, D., Weinstein, L.: Wikipedia Risks, in *Inside Risks* 186. Communications of the ACM 48(12) (2005)
4. Meyer, B.: Defense and Illustration of Wikipedia (2006), <http://se.ethz.ch/~meyer/publications/wikipedia/wikipedia.pdf> (visited January 15, 2009)
5. Orlowski, A.: Avoid Wikipedia, warns Wikipedia chief, It can seriously damage your grades. The Register (June 15, 2006)
6. Seigenthaler, J.: A false Wikipedia 'biography'. Editorial in USA TODAY (November 29, 2005)
7. Chittenden, M.: Comedy of errors hits the world of Wikipedia. The Sunday Times (February 12, 2006)
8. Giles, J.: Internet encyclopaedias go head to head. Nature (December 15, 2005)
9. Britannica, E.: Fatally Flawed. Refuting the recent study on encyclopedic accuracy by the journal Nature (March 2006)
10. Jensen, C.D.: Integrity in in Open Collaborative Authoring Systems. In: Proceedings of the Joint iTrust and PST Conferences on Privacy, Trust Management and Security (2007)
11. Amazon website, <http://www.amazon.com> (visited December 28, 2008)
12. eBay Internet auction website, <http://www.ebay.com> (visited March 12, 2009)
13. Against Intuition, Inc.: WOT website, <http://www.mywot.com/en/wot/home> (visited December 28, 2008)
14. Jøsang, A., Ismail, R., Boyd, C.: A Survey of Trust and Reputation Systems for Online Service Provision. *Decision Support Systems* 43(2), 618–644 (2007)
15. Zaihrayeu, I., da Silva, P.P., McGuinness, D.L.: IWTrust: Improving user trust in answers from the web. In: Herrmann, P., Issarny, V., Shiu, S.C.K. (eds.) *iTrust 2005*. LNCS, vol. 3477, pp. 384–392. Springer, Heidelberg (2005)
16. Dondio, P., Barrett, S., Weber, S., Seigneur, J.-M.: Extracting Trust from Domain Analysis: A Case Study on the Wikipedia Project. In: Yang, L.T., Jin, H., Ma, J., Ungerer, T. (eds.) *ATC 2006*. LNCS, vol. 4158, pp. 362–373. Springer, Heidelberg (2006)
17. Thomas Adler, B., de Alfaro, L.: A Content-Driven Reputation System for the Wikipedia. In: Proceedings of the 16th international conference on World Wide Web, pp. 261–270 (2007)
18. Kramer, M., Gregorowicz, A., Iyer, B.: Wiki Trust Metrics based on Phrasal Analysis. In: Proceedings of the 4th International Symposium on Wikis (WikiSym 2008), Porto, Portugal (2008)
19. Kittur, A., Suh, B., Chi, E.H.: Can you ever trust a wiki?: impacting perceived trustworthiness in wikipedia. In: Proceedings of the ACM 2008 Conference on Computer Supported Cooperative Work, pp. 477–480 (2008)
20. Douceur, J.: The Sybil Attack. In: Druschel, P., Kaashoek, M.F., Rowstron, A. (eds.) *IPTPS 2002*. LNCS, vol. 2429, pp. 251–260. Springer, Heidelberg (2002)
21. Back, A.: Hashcash – A Denial of Service Counter-Measure. Technical report (2002)
22. Dwork, C., Naor, M.: Pricing via Processing or Combating Junk Mail. In: Proceedings of Twelfth Annual International Cryptology Conference, pp. 139–147 (1992)
23. Clark, D., Wilson, D.: A comparison of commercial and military security policies. In: Proceedings of the IEEE Symposium on Security and Privacy, pp. 184–195 (1987)

24. Biba, K.J.: Integrity Considerations for Secure Computer Systems. Technical Report MTR-3153, The MITRE Corporation, Bedford, Massachusetts, U.S.A. (1977)
25. Bell, D.E., LaPadula, L.J.: Secure Computer Systems: Mathematical Foundations. Technical Report MTR-2547 (Volume I + Volume II), The MITRE Corporation (1973)
26. Fraser, T.: LOMAC: LowWater-Mark Integrity Protection for COTS Environments. In: Proceedings of the IEEE Symposium on Security and Privacy, pp. 230–245 (2000)
27. Thomas Adler, B., Chatterjee, K., de Alfaro, L., Faella, M., Pye, I., Raman, V.: Assigning Trust to Wikipedia Content. In: Proceedings of the 4th International Symposium on Wikis (WikiSym 2008), Porto, Portugal (2008)

# On Usage Control in Data Grids

Federico Stagni<sup>1</sup>, Alvaro Arenas<sup>2</sup>, Benjamin Aziz<sup>2</sup>, and Fabio Martinelli<sup>3</sup>

<sup>1</sup> INFN sez. di Ferrara, via Saragat 1 - 44100 Ferrara, Italy  
stagni@fe.infn.it

<sup>2</sup> e-Science Centre, STFC Rutherford Appleton Laboratory, Oxfordshire, UK  
{A.E.Arenas,B.Aziz}@rl.ac.uk

<sup>3</sup> IIT-CNR, via G. Moruzzi 1 - 56123 Pisa, Italy  
Fabio.Martinelli@iit.cnr.it

**Abstract.** This paper reasons on usage control in Data Grids. We adapt the UCON<sub>abc</sub> usage control framework for the case of distributed systems with multiple authoritative points. We call it the distributed usage control model. Then, we present an architecture implementing such model. In doing so, we use the functional components of the current Grids. Finally, we show a simple way for controlling the policy granularity using Semantic Grid technologies for the specification of policy subjects and objects.

## 1 Introduction

Data Grids [22] are an innovative technology taking advantage of existing computer science concepts in file systems, database systems and Grid computing. A Data Grid provides services that help users discover, transfer, and manipulate large datasets stored in distributed repositories and create and manage copies of these datasets. As a minimum, a Data Grid provides two basic functionalities: a high-performance reliable data transfer mechanism and a scalable replica discovery and management mechanism. However, as in any resource sharing environment, robust and rigorous treatment of data security in a Data Grid is vital. Moreover, since data is being shared over multiple administrative domains over the Grid, continuous monitoring and control of the data access is required.

At the present time, the majority of Data Grid middlewares and tools are growing behind some specific needs, mainly HEP (High Energy Physics) experiments. HEP applications produce and consume a considerably high amount of data with heavy impact on the bandwidth, but probably they don't need a high security system, because the main purpose of this activities is to be fast. At the same time, other Grid middlewares grow for chemicals or bioinformatics necessities, with different, tighter, security requirements.

A growing number of researchers and Virtual Organizations (VOs) will be born. They will use Grids, peer-to-peer systems, or whatever distributed paradigm will be in place that could help with their computing needs. These VOs may pose new security requirements. Just to make the simplest example, in the next generation of Grids file sharing, a user will want to give access to his/her files only to a limited set of people, identified by some kind of property. To do this, there's the need for a high control over who is authorized to view or modify the data [8].

Every Grid application may have a specific set of security requirements, and a Grid middleware should be capable to deal with a vast number of those. Different Grid applications should be able to determine the way the Grid guarantees data integrity and confidentiality. Different Grid authentication and authorization capabilities need to be in place. The solution is to conceive a really flexible system, with no explicit bindings with a specific application. To deal with these requirements, we apply usage control methodologies in a distributed security model, and apply it to a Data Grid abstraction.

This paper studies usage control techniques for Data Grids. Usage control extends traditional access control by controlling data access as well as usage [17][15]. Recently there has been a fresh interest in applying usage control to Grid systems [10][25]. We develop here a usage control model suitable for multi-authoritative distributed systems. We base this model on the  $UCON_{abc}$  model proposed by Park and Sandhu [14]. The main contribution of the paper is a Data Grid usage control architecture using the functional components of the current Grids, as presented by the Open Grid Forum (OGF) group on Grid authorization. We also consider the advantages of using Semantic Grid technologies for the specification of UCON subjects and objects for controlling the policy granularity.

The rest of the paper is structured as follows. Section 2 introduces an abstraction of Data Grids and some terminology. In Section 3, we give some background on UCON and introduce the distributed usage control model. Section 4 shows the proposed Data Grid usage control architecture. In Section 5 we reason how to take advantage of Semantic Grid technologies for controlling the policy granularity. Finally, Section 6 discusses related work, and Section 7 concludes the paper and highlights directions for future works.

## 2 An Abstraction of Data Grids

A distributed system may contain a variety of data resources. These resources may use different data models to structure the data, different physical media to store it, different software systems to manage it, different schema to describe it, and different protocols and interfaces to access it. The data may be stored locally or remotely; may be unique or replicated; may be materialized or derived on demand. Different levels of virtualizations over these data resources should be provided. Virtualizations provide abstract views that hide these distinctions and allow the data resources to be manipulated without regard to their nature. The Data Grid abstraction we provide here helps us with future reasonings.

In a Data Grid there are two kinds of resources to be managed: *Grid Data* and *Grid Storage Space*. A *Grid Data* (GD) is any kind of data that can be located, transferred, replicated and manipulated: client services should be able to access a dispersed GD, independently from its physical location, through a *Data Grid Management System* (DGMS) [12]. A DGMS is a software system used to manage Data Grids through the use of multiple abstraction mechanisms that hide the complexity of distributed data and heterogeneous resources. This naming capability allows users to refer to specific data resources in a physical storage system using a high level logical identifier. A *Grid Storage Space* (GSS) is a storage space shared between multiple VOs, and managed by a *Grid Storage Element* (SE). An SE (e.g. the Storage Resource Manager [7]) is an



interface to mass storage systems, providing a uniform control interface and enabling the Grid to efficiently use the storage.

It is not necessary for a GD to be stored in a GSS only, while a GSS may also contain data that cannot be relocated, viz. are not GD. We are not interested in the security implications of non-GD data.

DGMS implementations should follow the OGF recommendations for providing implementation guidelines and standards to implement GD location independence. Data resources have to be recognized by name without any location information. The Open Grid Services Architecture (OGSA) work on data architecture [11] identifies a scheme with the following three levels of naming:

- **Human-Oriented Name (HON):** Based on a naming scheme that is designed to be easily interpreted by humans, viz. human-readable and human-parsable. The HONs are user friendly high-level identifiers by which the users find the actual locations of their files. The same data resource could be addressed by various HONs by different users, similarly to the concept of alias. A number of HONs can be mapped to a single *Abstract Name*.
- **Abstract Name (AN):** A persistent name suitable for machine processing that does not necessarily contain location information. ANs are given to each data managed by a DGMS. An AN is a unique identity to hide the data replication: the same AN can correspond to different replicas.
- **Address:** Specifies the location of a data resource. An address provides an abstraction of the data namespace living into a storage resource to allow different data access paths. Each replica has its own address and it specifies implicitly which storage resource needs to be contacted to extract the data.

Figure 1 shows a simplified logical view of a Data Grid. We distinguish two kinds of data accesses: (i) clients (e.g. Grid users) access a GD knowing just the HON by

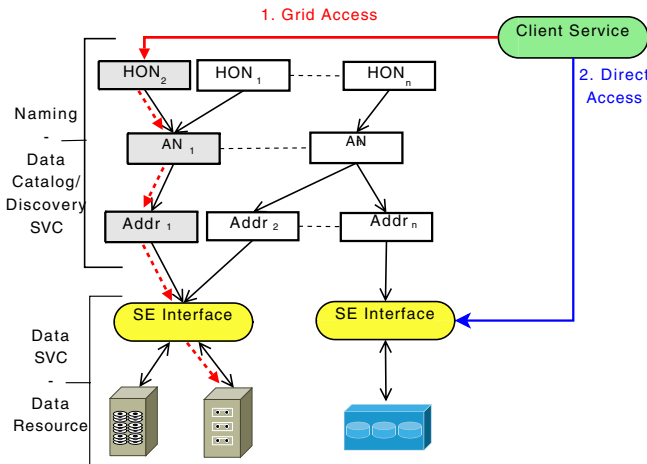


Fig. 1. A logical view of a Data Grid

performing what we call a *Grid access*, and (ii) access GD and non-GD data directly on the SE when the address is known, thus performing what we call a *direct access*.

### 3 A Distributed Usage Control Model

In this section we define a usage control model for Grids and distributed systems. The model can use the  $UCON_{abc}$  usage control model for the specification of usage control policies. We chose  $UCON_{abc}$  because of its high capabilities, as will be clear by reading below. Then, we'll apply this model to the Data Grid abstraction of Section 2.

#### 3.1 $UCON_{abc}$

The main novelty of the UCON model lies in the fact that subjects and objects may have attributes that are mutable thereby facilitating the continuity of the decision making and policy enforcement processes. Additionally, while decisions in access control models are usually based on *authorizations* only, the UCON model introduces two other decision factors, namely *obligations* and *conditions*. All of these features render the UCON model attractive for specifying security policies in Data Grids, especially considering the plethora of various security needs coming from the different Data Grid applications.

The UCON model comprises the following elements:

- *Subjects, Objects and Rights*: the subject is the entity that exercises rights, i.e. that executes usage operations on objects. An object, instead, is an entity that is accessed by subjects through access operations. Rights are the privileges that subjects can exercise on objects. Traditional access control systems view rights as static concepts, for instance access matrices, which do not change over time or have a slow rate of change. Instead, UCON determines the existence of a right dynamically, whenever a subject attempts to use and exercise a right on some object. Hence, if the same subject accesses the same object several times, the UCON policy could grant the subject different access rights each time based on changing attributes of the subject and/or the object.
- *Attributes*: both subjects and objects have attributes. These attributes can be *mutable*, i.e. they can change over time, or *immutable*, i.e. they are constant over time. An example of a mutable attribute is the number of times that a subject accesses an object, whereas an immutable is a subject's or an object's identity. Conditions can use attributes representing the system status which are not under the UCON service control.
- *Predicates*: predicates are logical statements about the subjects' and objects' attributes and the requested right. Predicates can be either *authorization*, *obligation* or *condition* predicates or any combination of these. Authorization predicates express a set of rules that determine whether to grant the requested right or not. The authorization predicates could exploit both attributes of the subject and of the object. The evaluation of the predicates can be performed before and during the execution of an action. Obligations are UCON decision factors that are used to verify whether subjects have satisfied, or continuously satisfy, some mandatory requirements before (during) an usage. Finally, conditions are environmental or system-oriented

decision factors that do not depend on subjects or objects. Conditions are evaluated at runtime when the subject attempts to perform the usage, before or during an action [26].

UCON<sub>abc</sub> is a family of models with several parameters. The presence of Authorizations (A), obligations (B) and Conditions (C), pre- and on-going decisions, as well as the mutability of attributes (immutable (0), preUpdate (1), onUpdate (2), postUpdate (3)) are the factors to be considered. For example, a PreA<sub>0</sub> policy is an pre-authorization policy with no attributes update, while an OnB<sub>3</sub> is a on-obligation policy with a postUpdate of one or more attributes, and so on. The various UCON models differ in the presence of attribute updates and in the sequentiality of the operations. Therefore, an enforcing mechanism for UCON policies should be able to enforce not only the single operations, but the sequence these operations are invoked. The different actions that subjects and system can perform in the UCON model relate to the different phases of an object's usage.

Given that the triple  $(s, o, r)$  represents the subject  $s$  requesting the right  $r$  for accessing the object  $o$ , we consider the following set of actions, which we borrowed from [26]: (i) TryAccess  $(s, o, r)$ : performed by subject  $s$  when performing a new access request  $(s, o, r)$ , (ii) PermitAccess  $(s, o, r)$ : performed by the system when granting the access request  $(s, o, r)$ , (iii) DenyAccess  $(s, o, r)$ : performed by the system when rejecting the access request  $(s, o, r)$ , (iv) the operation RevokeAccess  $(s, o, r)$  is performed by the system when revoking an ongoing access  $(s, o, r)$ , (v) EndAccess  $(s, o, r)$ : performed by a subject  $s$  when ending an access  $(s, o, r)$ , and (vi) AttributeUpdate  $(s, o, r)$ : performed by the system to update a subject or an object attribute when performing an access request  $(s, o, r)$ .

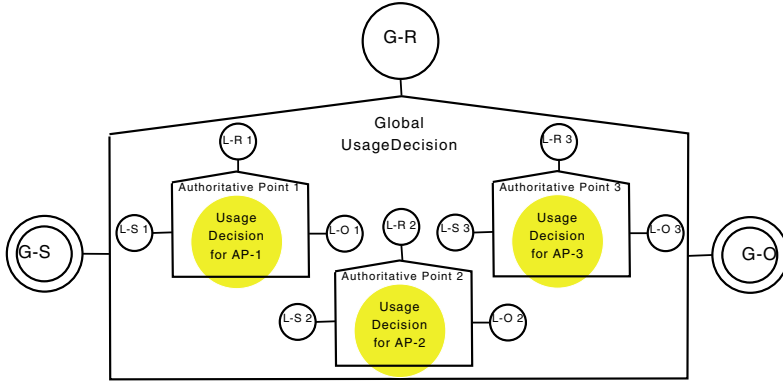
All the policies pertaining to the UCON authorization, obligation and condition core models are defined for positive permissions: if there is no policy to enable the permission according to the attribute values, then the usage is denied by default. This is sometimes called the closed system assumption, whereby no policy is specified to deny an access in a system.

### 3.2 The Distributed Usage Control Model

Up to now, there's no existing security model that can cope with the inner nature of Grids. In a distributed system like a Grid, there may be small to larger number of different resources, each one controlled by a different policy officer. Each policy officer is a *Source of Authority* (SoA) for an *authoritative point*, viz. authoritative sources of authorizations and usage control. When a client service is requesting the permission to access a single remote resource, a number of policies maintained by different SoAs may have to be evaluated. This requirement was historically advocated by the Globus and EGEE (Enabling Grids for E-science) security teams [20] and, up to now, there is no existing Grid usage control framework coping with this requirement. Therefore, the challenge for controlling the resource usage in Grids and distributed systems is knowing which are the authoritative points involved in a usage request. The model we propose in this Section can deal with such requirement.

Within the model we propose, that we call *Distributed Usage Control Model* (D-UCM), policy officers could impose the evaluation of local policies. We say that a single

usage decision comes from the evaluation of a *workflow* of local usage control steps. For example, when the workflow of a complete usage control is made of three separate usage control steps, each one of the three must be satisfied. If one of the usage control steps can't be satisfied, the entire usage is not permitted.



**Fig. 2.** The Distributed Usage Control Model

Figure 2 shows a pictorial overview of D-UCM. Within this Figure, we show that three distinct authoritative points each impose the evaluation of a local usage decision (L-UD) step. Each step has to be satisfied for the enforcing of a global usage decision (G-UD). A central workflow orchestrator, with responsibility for the G-UD, is needed. The evaluation of a L-UD step is seen as an atomic action. The model doesn't pose any constraint neither on the way authoritative points enforce usage control steps, nor on the nature of the security policies that have to be evaluated to reach a L-UD. For example, a L-UD may require the evaluation of a vast number of distributed and concurrent policies, but all this machinery is under the responsibility of the local Source of Authority (SoA).

A G-UD is based on a global subject (G-S), a global object (G-O) and a requested global right (G-R). To reach a L-UD, each SoA encodes G-S, G-O and G-R respectively in a local subject (L-S), local object (L-O) and local right (L-R). The relation between the global and local subjects, objects and rights is dependent from the application using the model.

## 4 Usage Control in Data Grids

Policy-based security mechanisms adopt an almost standard terminology when defining authorisation architectures, which distinguishes between different kinds of *Policy Points*. Their definition has strong connection with traditional access control techniques. In this paper, we continue using the same terminology when drawing a distributed usage control architecture for Data Grids. In doing so, we use functional components defined by the OGSA-Authz GWD-I draft architecture for a Grid service provider authorisation service middleware [3].

In the OGSA work, great attention is put on *credentials*, defined as attribute assertions digitally signed by the issuer (i.e. a security token) so that it can be cryptographically validated. Credentials can be issued by the Credential Issuing Services (CISs) of an Identity Provider or an Attribute Authority (e.g. the Virtual Organization Membership Service (VOMS) [21]). Credentials can then be validated by a Credential Validation Service (CVS), that return the valid attributes of the subject. A Policy Decision Point (PDP) is the component responsible for returning an authorization decision given the user's access request and the user's valid attributes. The Policy Enforcement Point (PEP) enforces the results returned from a policy engine (normally a PDP). The Context Handler (CH) is responsible for handling the communications between PEPs, CVSs and PDPs. The interactions between these functional components can be constructed in four different ways, according to whether the credentials and the authorization decisions are pulled or pushed. For example, Figure 3 shows the case where an access requestor (a Grid User) pushes his/her credentials to a PEP. Then, after the CH obtained valid attributes from the CVS, a PDP is interrogated for an authorization decision, which in the end is returned to the PEP.

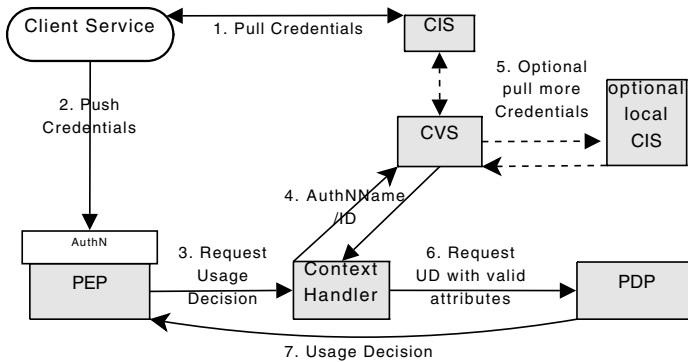


Fig. 3. OGSA functional components

#### 4.1 A Data Grid Usage Control Architecture

We now apply the *distributed usage control model* (D-UCM) of Section 3.2 to the Data Grid. The reason we chose  $UCON_{abc}$  as a policy model is because it encompasses traditional access control models, and does not pose constraint on the degree/level of granularity of usage control, ranging from storage space level to individual data access restrictions.

We now consider the terminology introduced in Section 2. In a Data Grid, GDs (Grid Data) are stored (and transferred and replicated) in GSSs (Grid Storage Spaces) by the SEs (Storage Elements). A client performing an access to a GD should be authorized to access the data itself, and to use the GSS. Therefore, the policies of the single steps should be written by those policy officers which are SoA for the GDs (e.g. VO admins

or simply VO participants), and by those policy officers which are SoA for the GSSs (e.g. SE admins). Therefore, we identified a couple of authoritative points, which are DGMS and SEs. A Complete usage control in Data Grid then follows a two-steps workflow. From now on, we refer to each of these steps as *data usage control* (D-UC) and *storage usage control* (S-UC). Each step corresponds to the enforcing of (at least) a  $UCON_{abc}$  policy.

We now pose some constraints on the relation between G-S, G-O, G-R and L-S, L-O and L-R of D-UC and S-UC. Each single L-S represents the G-S as it is recognized by respectively the DGMS and the SE. Similarly, the L-R represents the G-R as it is recognized by the DGMS and by the SE. The object of the D-UC is the unique identifier of a GD, i.e. the *abstract name*. The object of S-UC is, instead, the GSS itself. By doing this neat separation between the objects of D-UC and S-UC, we highlight the role of the authoritative points. Moreover, by doing this separation, the policies of the different steps will never overlap. Figure 4 shows this two-step usage control.

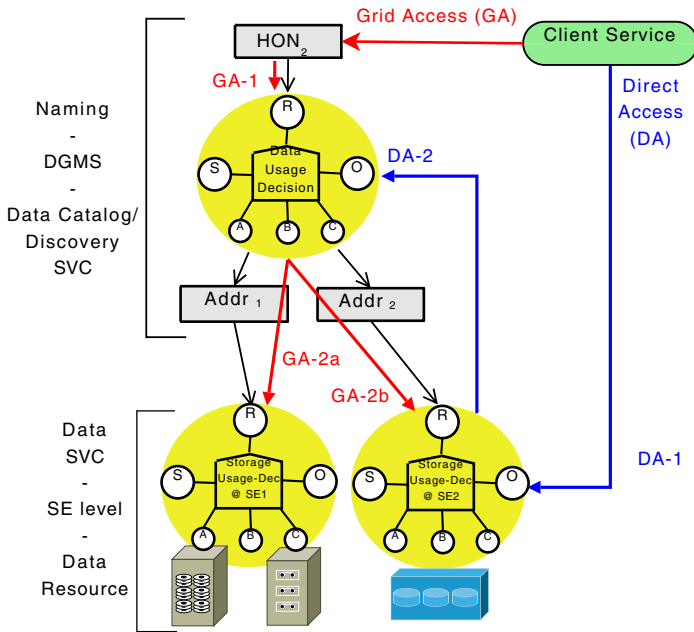


Fig. 4. The two-steps Data Grid usage control

There are many differences between existing security models for Grids and the one we are proposing, but the most apparent one can be seen in Figure 5. The Figure shows security models for Data Grids as seen in [6] with the D-UCM for Data Grids (on the right). The main difference stands with the usage of two (UCON) PDPs, one for each usage control step.

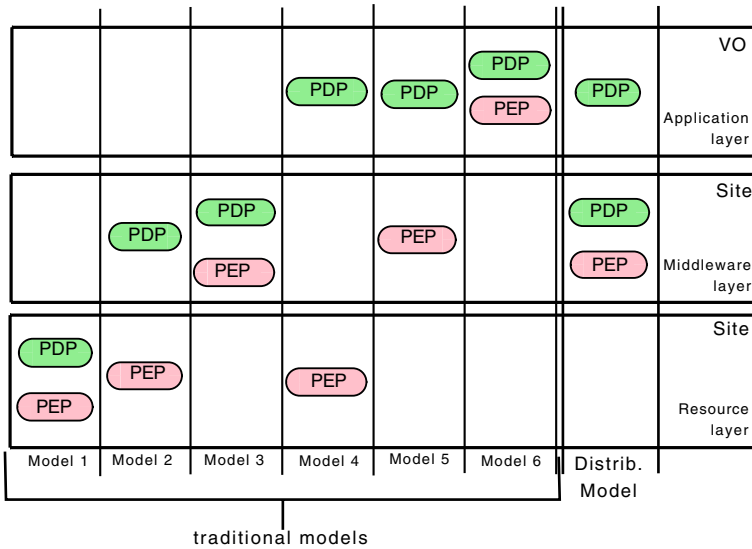


Fig. 5. Comparison of security models for Data Grids (inspired from [6])

Figure 6 shows a Data Grid security architecture implementing the D-UCM on Data Grids. Each usage control step uses the authorization functional components defined by OGSA. A *Client Service* is an access requestor (normally, a Grid User) that pushes the credentials obtained from a VO CIS either to a DGMS (when performing a Grid access) or to a SE (when performing a Direct Access). DGMS and SE are clients to a *super-PEP* software element, which communicate with the CHs located at the DGMS and SEs. Each CH obtains valid attributes from the CVS. Then, the local UCON PDP is interrogated for an authorization decision. A UCON PDP should be capable to interpret, i.e. enforce, policies pertaining to the UCON<sub>abc</sub> usage control framework. The UCON PDP is responsible for returning an usage decision to the super-PEP, given the user's usage request (i.e. the right requested), the user's valid attributes, the object's valid attributes, and the satisfaction of authorizations, obligations and conditions predicates. From a UCON point of view, valid attributes released by a CVS are examples of *immutable* (persistent) attributes.

The *super-PEP* is the software element responsible for performing both the usage control steps requested. Among the possible solutions for this element, a centralized service or a collaborative one. For instance, one could consider POLPA [11], a policy language suitable for expressing sequence of actions as well as conjunctions and disjunctions of such sequences. These policies could be useful to *orchestrate* other usage control steps in a workflow (as well as to model single access actions in a usage control step). A possible initial solution in this line of thought is envisaged in [2]. Due to the fact that a super-PEP may be located at DGMS or at SE level, we consider it as a mobile agent.

A complex UCON PDP should be able to evaluate policies where the predicates are statements about the subjects' and objects' attributes. Five sub-components make up the UCON PDP:

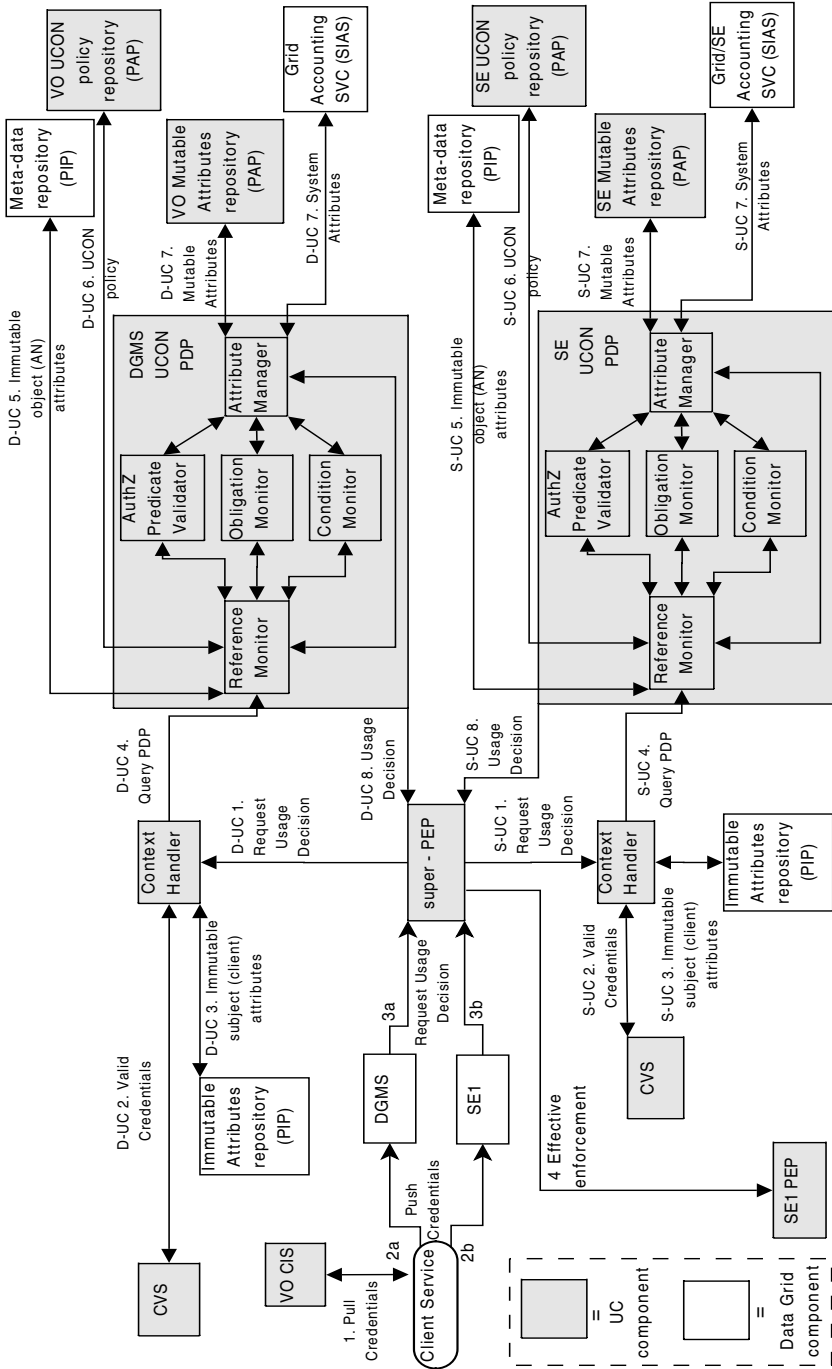


Fig. 6. Data-Grid usage control architecture



- the **Reference Monitor** (RM) is a gateway for all the usage decisions; it can receive `TryAccess` and `EndAccess` invocations, and is responsible for issuing the `PermitAccess`, `DenyAccess` or `RevokeAccess` operations;
- the **Authorization Predicate Validator** (PV) takes care of validating the authorization policy predicates; it can be perform the `AuthzPredicateValidation` operation;
- the **Obligation Monitor** (OM) checks if subject fulfilled the obligations; it can be perform the `ObligationsSat` operation;
- the **Condition Monitor** (CM) takes care of validating the condition policy predicates; it can be perform the `CondsPredicateValidation` operation;
- the **Attribute Manager** (AM) updates the UCON mutable attributes and return their values; it can perform the `AttributeUpdate` operation.

External components are needed to supply the UCON PDP with the needed information: (i) an *UCON policy repository* provides the PDP with the UCON policies to be evaluated, (ii) a *meta-data repository* provides the PDP with the optional immutable object attributes, (iii) a *mutable attributes repository* stores the UCON mutable attributes of the subjects and objects, and (iv) the *Grid/SE Accounting SVC* is a System Information/Accounting Service (SIAS) acting as a source for system attributes. For an access, the PDP collects the immutable subject and object attributes, as well as search for the UCON policies to be enforced. The policy is selected using the the UCON subject and object requested. Mutable subject and object attributes, as well as system attributes, are pulled by the PDP from the mutable attribute repository, and from the Grid accounting service.

For what regards the *data usage control* step, the rights are defined at the level of the *abstract name*. Thus, we apply the following restrictions: (i) an **UCON subject** is represented by a DGMS user ID, which is the way the access requestor Grid user ID is recognized by the DGMS; (ii) an **UCON object** is represented by the *abstract name* requested by the DGMS user ID; (iii) an **UCON right** always follows in one of the fundamental rights categories, which are *view* (read) and *modify* (write), possibly augmented with *creation* and *deletion*; (iv) **subject attributes** are mutable or persistent security descriptors of the *Client Services* (e.g. the number of data accessed); (v) **object attributes** are mutable or persistent security description of the *abstract name* (e.g. the *privacy* level, or the maximum number of contemporary access);

Insted, since the *storage usage control* step defines rights at the address level, we apply the following restrictions: (i) an **UCON subject** is an SE user ID, which is the way the access requestor Grid user ID is recognized by the SE; (ii) an **UCON object** is the GSS where the GD is located; (iii) an **UCON right** depends from the SE interface implementation in use; (iv) **subject attributes** are security descriptors of the *Client Services*; (v) **object attributes** are security descriptors of the GSS;

## 4.2 Architecture Analysis

### Main Pros

- The whole architecture is **modular**, **flexible**, and presents a **high capability** level. A number of policy officers are capable of specifying policies pertaining to a vast

number core models, and these policies will never overlap. Moreover, each SoA maintain a local authority over its resources, and there's no need for policy synchronization.

## Main Cons

- **Complexity.** The proposed architecture has a high degree of complexity. We are aware of the fact that *Complexity is the worst enemy of security*.<sup>1</sup> There are reasons for such complexity, and simplification possibilities. All the software elements composing the UCON PDP have been recognized as requirements for enforcing UCON<sub>abc</sub> policies. To do so, we used notions that are partially extracted from the KAOS requirement engineering methodology to produce an abstract specification of all the UCON PDP architectural elements and operations. Such work is partially available in [18]. We also demonstrated that such specification is capable to enforce all the UCON<sub>abc</sub> types of policies, as they are formally specified in [26].  
An overall simplification is possible: since UCON is a family of core models, simpler UCON PDPs would enforce not all, but a number of UCON core models. For example, the *Obligation Monitor* component is not necessary if there are no needs for enforcing UCON<sub>b</sub> policies.
- **Performance and Trust.** Other big problems may be represented by the performance of an implementation, and by the trust relationships between the sites, but since right now there's not a single complete implementation of the architecture, we leave this problem to future works on the topic.

## Issues

- **Policy strategy.** Near the end of Section 3, we mentioned that an enforcement mechanism for UCON policies should be able to enforce not only the single operations, but the sequence these operations are invoked. In order for an UCON PDP to be an enforcement mechanism for all the UCON core policy models, a way to encode the policy *strategy* (i.e. the sequentiality of the operations) is needed. A possibility lies in the use of an operational policy language like the already cited PoLPA, where the policy specification itself encodes the strategy. Otherwise, an external scheduler can be used for the particular UCON<sub>abc</sub> sub-model to which the policy pertains.
- **Obligations.** Checking the obligations satisfaction is still an issue. An introductory work on usage control obligations can be found in [16]. We don't plan to solve such issue within this paper.

We believe this concrete architecture can be of real use for implementors and developers.

## 5 Usage Control in Semantic Grids

As stated in [4], the Semantic Grid is an extension of the Grid in which rich resource metadata is exposed and handled explicitly, and shared and managed via Grid

<sup>1</sup> Bruce Schneier, Crypto-Gram newsletter, March 2000.

protocols. The layering of an explicit semantic infrastructure over the Grid infrastructure potentially leads to increased interoperability and greater flexibility.

In the near future, data on the order of hundreds of petabytes will be spread in multiple storage systems worldwide dispersed in, potentially, billions of replicated data items. The creation, definition and enforcement of usage control policies may represent an issue in terms of management, scalability, governability and consistency. For example, in current hierarchical file systems, access control is made specifying the authorizations on every one of billions of files. If usage and access control techniques are to be really useful in a large pervasive environment, they should be able to solve the scalability and governability problems presented by the more traditional access control models, such as Identity Based Access Control (IBAC) — normally implemented using Access Control Lists (ACLs) — or even the more flexible Role Based Access Control (RBAC) [5]. In the implementations of traditional access control models, when an authorization policy changes for a specific user or role, the security manager must implement the adjustment in every entry involved, potentially all. These factor may generate a policy explosion phenomenon. What's needed is a mechanism for keeping under control the policy granularity. A simple solution lies in the semantic binding assertions regarding Grid users and resources, as exposed in a Semantic Grid. UCON subjects and objects may be semantic concepts extracted from those VO ontologies or scientific model ontologies used in the Semantic Grid.

Before going any further, we make a clear distinction between semantic attributes and UCON attributes. Semantic attributes can globally describe users, data and resources properties, but are not meant to be security attributes. Instead, the UCON attributes define only subjects' and objects' security properties, and for many of them there is no need to be known outside the usage control service. In a Semantic Grid, following the terminology introduced in [4], each *Grid Entity* is associated to a *Knowledge Entity* (KE) through a *Semantic Binding*. KEs are special types of Grid Entities that represent or could operate with some form of knowledge. Examples of KEs are ontologies, rules, knowledge bases or even free text descriptions that encapsulate knowledge that can be shared. Semantic Bindings are the entities that come into existence to represent the association of a Grid Entity with one or more KE.

A semantic-aware UCON PDP is depicted in Figure 7 and is obviously much similar to the one presented with Figure 6. In a Semantic Grid, the client service (i.e. the Grid User) and the data to be accessed (e.g. the abstract name managed by the DGMS) are represented by a KE. For what concerns the DGMS, the metadata repository can be used to store the KE of the abstract names. Even if in Semantic Grids specific Grid Users will keep asking to access specific Grid Data, a semantic-aware PDP would search for applicable policies using the multiple fields of the KEs of both the Grid user and the resource to be accessed. In this way, two or more policies could be applicable for a single access request, thus generating more than a single policy control for a single access request. When no policy is applicable, the access is denied. When multiple UCON Pre{ABC} policies are to be evaluated, even if just one is satisfied, then the access has to be permitted. When multiple UCON On{AB} policies are to be evaluated, even if just one is no more satisfied, then the access will be revoked.

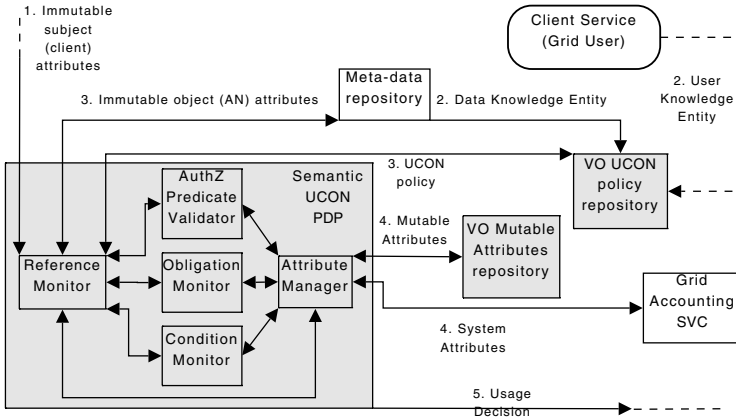


Fig. 7. A semantic-aware UCON PDP for Semantic Grids

Example of KEs representing the Grid Entity *Grid User* and the GD are shown in Figure 8. A semantic-aware DGMS could associate a data KE like this one to each of the managed *abstract names*. The Grid User KE graph is inspired from [4], while GD graph has been derived from the CCLRC scientific metadata model [19]. These examples are not meant to be complete. Each Grid User is simply described through the use of three fields: the *Institution* he/she is affiliated with, the *Investigation* he/she takes part in, and the *Job or Role* he/she is doing as part of the *Institution*. Instead, each GD is described not only by the *Type* (e.g. file, or stream), but also by the *Program* of work, the supported *Study*, and by an *Investigation*.

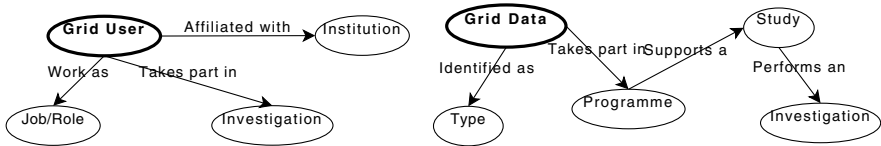


Fig. 8. An example for a Grid user and a GD Knowledge Entity

A security administrator can control the policy granularity using the semantic fields shown in Figure 8 for the definition of collective policies, like the following simple PreA<sub>0</sub> policy (written in POLPA, where “.” represents sequence of actions):

- 1 TryAccess(Institution:STFC, Study:ISIS, read).
- 2 PredicateValidation([]).
- 3 PermitAccess(Institution:STFC, Study:ISIS, read).
- 4 EndAccess(Institution:STFC, Study:ISIS, read).

This policy states that each User associated with the Institution STFC can read those GD pertaining to the ISIS study. UCON attributes can be associated to these UCON subjects and objects.

With this simple approach, it's easy to realize a fuzzy security [24] for Grids. The possibility to control the policy granularity, and thus to avoid the policy explosion is of particular interest for those VOs that consider the specification of a per-user, per-role or per-data policies a useless effort. High Energy Physics VOs usually fall in this category.

## 6 UCON Implementations

We are not aware of specific usage control frameworks for Data Grids, although there are already some running implementations for other scenarios.

In [10], Martinelli and Mori provide a model for usage control for computational Grids for the Globus Toolkit, following Sandhu's UCON model. The prototype implements the standard PEP-PDP architecture, and the PoLPA policy language is used to encode UCON policies. The PEP has been integrated within the application execution environment to monitor the accesses to the local resources performed by the applications executed on behalf of remote GRID users. The PDP gets the security policy from a repository, and builds its internal data structures for the policy representation. The PDP is invoked by the PEP every time the subject attempts to access a resource. It exploits its representation and determine whether the access should be allowed or not, returning to the PEP `permit` and `deny` invocations. The PDP continuously evaluates a set of given authorizations, conditions and obligations while an access is in progress, and it could invoke the PEP to terminate it through a `revoke` action. The architecture comprises the managers for attributes, conditions and obligations. The *Condition Manager* is invoked by the PDP every time the security policy requires the evaluation of a condition. The *Attribute Manager* is in charge of retrieving and updating the value of attributes. The *Obligation Manager* monitors the execution of obligations. Martinelli and Mori focussed on single GRID computational services. We argue that the adaptation of UCON to Data Grid poses a greater number of issues to be solved. This paper highlighted a number of them.

In [25], Zhang *et al* propose a UCON prototype implementation. The security architecture leverages a centralized attribute repository in each VO and a usage monitor in each Resource Provider (RP) for attribute management. The policies are specified with the eXtensible Access Control Markup Language (XACML) [13], which, as recognized by the same authors, seems suffers of several limitations to exactly encode UCON policies. Both PDP and PEP are located on the RP side. For an access, the PDP collects the subject, object and system attributes, and makes the usage control decision, which is enforced by the PEP. The immutable subject attributes are pushed to the PDP by the requesting subject. This prototype has not been applied to an actual (Data) Grid security architecture, like the OGSA one.

Due to increasing number of kernel-level attacks The protection of the kernel integrity is one of the most essential security goal in building a trustworthy operation system. An approach based on UCON model for Linux kernel protection was proposed at [23].

Pioneer works, specifying usage control requirements with mobile and ubiquitous computing application, were presented at [9].

Even if these prototypes should be considered when implementing a Data Grid usage control architecture, none of them consider the inner multi-authoritative nature of Data

Grids and their specific issues. We plan to implement our architecture and check its feasibility and performances for real applications by starting from these experiences.

## 7 Conclusion and Future Work

Different Grid applications running on the same middleware may need different security levels. A middleware security service should be modular and flexible, in order to accommodate disparate Grid applications authorization requirements. We believe that usage control techniques, as presented in this paper, are a step toward the right direction. We proposed a usage control model for Grids and distributed systems that uses a *work-flow* of usage control steps. Each step implements a distinct usage control through the enforcement of at least a  $UCON_{abc}$  policy. In a Data Grid, a complete usage control is performed with two separate steps. Then, we presented a flexible distributed usage control architecture for Data Grids with a strong reference to the OGSA work on Grid authorization architecture. We also showed a simple way of using the Semantic Data-Grids KEs for controlling the policy granularity, thus avoiding the policy explosion phenomenon.

We consider this paper as a step toward an integrated usage control framework for Data Grids. We believe that many of the ideas presented here can be adapted for the case of computational Grids and distributed systems alike. Regardless of it, there is still issues to be solved. Some of them have been highlighted in Section 4.2. We are currently analysing deployed policy languages and authorisation mechanisms in order to determine their capacity to implement  $UCON_{abc}$  policies as presented here, looking at the possibility of extending one of the already developed implementations. The final goal is to either propose a new implementation, or extensions to the already developed ones. Works in these directions have already started.

*Acknowledgements.* This work is partially funded by the EU CoreGRID project, contract No. 004265, and the EU GridTRUST project, contract No. 033827.

## References

1. Antonioletti, M., Berry, D., Chervenak, A., Kunszt, P., Luniewski, A., Laws, S., Morgan, M.: Ogsa data architecture v0.6.6. Technical report, Open Grid Forum (2007), <http://forge.gridforum.org/sf/go/doc13635?nav=1>
2. Aziz, B., Arenas, A., Martinelli, F., Matteucci, I., Mori, P.: Controlling usage in business process workflows through fine-grained security policies. In: Springer (ed.) 5th International Conference on Trust, Privacy & Security in Digital Business (2008)
3. Chadwick, D.: Functional components of grid service provider authorisation service middleware. Technical report, Open Grid Forum (2008), <http://forge.gridforum.org/sf/go/doc15171?nav=1>
4. Corcho, Ó., Alper, P., Kotsiopoulos, I., Missier, P., Bechhofer, S., Goble, C.A.: An overview of s-ogsa: A reference semantic grid architecture. J. Web Sem. 4(2), 102–115 (2006)

5. Ferraiolo, D., Sandhu, R., Gavrila, S., Kuhn, D., Chandramouli, R.: Proposed nist standard for role-based access control. *ACM Transactions on Information and System Security (TIS-SEC)* (3), 224–274 (2001)
6. Frohner, A., Kunszt, P.Z., Brito da Rocha, R., Laure, E.: Security of distributed data management. Technical Report EGEE-TR-2006-003. EGEE-TR-2006-DATASEC (2006)
7. Group, T.S.R.M.W.: An internet attribute certificate profile for authorization (2008), <http://sdm.lbl.gov/srm-wg/doc/SRM.v2.2.pdf>  
<http://sdm.lbl.gov/srm-wg/doc/SRM.v.2.2.pdf>
8. HealthGrid: Healthgrid white paper. Technical Report HealthGrid-White\_Paper-Draft\_v.1.1-5, HealthGrid (2004)
9. Hilty, M., Pretschnner, A., Schaefer, C., Walter, T.: Usage control requirements in mobile and ubiquitous computing applications. In: *ICSNC 2006: Proceedings of the International Conference on Systems and Networks Communication*, p. 27. IEEE Computer Society, Los Alamitos (2006)
10. Martinelli, F., Mori, P.: A Model for Usage Control in GRID systems. In: *Grid-STP 2007, International Conference on Security, Trust and Privacy in Grid Systems*. IEEE Computer Society, Los Alamitos (2007)
11. Martinelli, F., Mori, P., Vaccarelli, A.: Towards continuous usage control on grid computational services. In: *ICAS/ICNS*, p. 82 (2005)
12. Moore, R., Jagatheesan, A., Rajasekar, A., Wan, M., Schroeder, W.: Data Grid Management Systems. In: *Proceedings of the 21st IEEE/NASA Conference on Mass Storage Systems and Technologies*, Maryland, USA (2004)
13. OASIS: Oasis extensible access control markup language (xacml) tc (2005), <http://www.oasis-open.org/committees/xacml>
14. Park, J., Sandhu, R.: The UCON<sub>abc</sub> Usage Control Model. *ACM Transactions on Information and System Security* 7(1), 128–174 (2004)
15. Pretschnner, A., Hilty, M., Basin, D.: Distributed usage control. *Communications of the ACM* (2006)
16. Pretschnner, A., Massacci, F., Hilty, M.: Usage control in service-oriented architectures. In: Lambrinouidakis, C., Pernul, G., Tjoa, A.M. (eds.) *TrustBus 2007*. LNCS, vol. 4657, pp. 83–93. Springer, Heidelberg (2007)
17. Sandhu, R.S., Park, J.: Usage control: A vision for next generation access control. In: Gorodetsky, V., Popyack, L.J., Skormin, V.A. (eds.) *MMM-ACNS 2003*. LNCS, vol. 2776, pp. 17–31. Springer, Heidelberg (2003)
18. Stagni, F., Arenas, A.E., Aziz, B.: On usage control in data grids. Technical Report TR-0154, Institute on Knowledge and Data Management, CoreGRID - Network of Excellence (2008)
19. Sufi, S., Matthews, B.M.: The cclrc scientific metadata model: a metadata model for the exploitation of scientific studies and associated data. In: *Knowledge and Data Management in Grids (2005)*, <http://epubs.cclrc.ac.uk/work-details?w=34195>
20. team, E.J.: Egee global security architecture for web and legacy services. deliverable EGEE-JRA3-TEC-487004-DJRA3.1-v1-1, EGEE JRA3 (2004)
21. Venturi, V., Stagni, F., Gianoli, A., Ceccanti, A., Ciaschini, V.: Virtual organization management across middleware boundaries. In: *E-SCIENCE 2007: Proceedings of the Third IEEE International Conference on e-Science and Grid Computing*, pp. 545–552. IEEE Computer Society, Washington (2007), <http://dx.doi.org/10.1109/E-SCIENCE.2007.84>
22. Venugopal, S., Buyya, R., Ramamohanarao, K.: A taxonomy of data grids for distributed data sharing, management, and processing. *ACM Comput. Surv.* 38(1), 3 (2006), <http://dx.doi.acm.org/10.1145/1132952.1132955>

23. Xu, M., Jiang, X., Sandhu, R., Zhang, X.: Towards a vmm-based usage control framework for os kernel integrity protection. In: SACMAT 2007: Proceedings of the 12th ACM symposium on Access control models and technologies, pp. 71–80. ACM, New York (2007), <http://doi.acm.org/10.1145/1266840.1266852>
24. Yao, D.: An ad hoc trust inference model for flexible and controlled information sharing. In: Security and Management, pp. 555–561 (2008)
25. Zhang, X., Nakae, M., Covington, M.J., Sandhu, R.: Toward a usage-based security framework for collaborative computing systems. *ACM Trans. Inf. Syst. Secur.* 11(1), 1–36 (2008), <http://doi.acm.org/10.1145/1330295.1330298>
26. Zhang, X., Parisi-Presicce, F., Sandhu, R., Park, J.: Formal Model and Policy Specification of Usage Control. *ACM Transactions on Information and System Security* 8(4), 351–387 (2005), <http://doi.acm.org/10.1145/1108906.1108908>



# Detection and Prevention of Insider Threats in Database Driven Web Services

Tzvi Chumash and Danfeng Yao\*

Rutgers University, Computer Science Department,  
110 Frelinghuysen Road, Piscataway, NJ 08854, USA  
{tzvika, danfeng}@cs.rutgers.edu

**Abstract.** In this paper, we take the first step to address the gap between the security needs in outsourced hosting services and the protection provided in the current practice. We consider both insider and outsider attacks in the third-party web hosting scenarios. We present *SafeWS*, a modular solution that is inserted between server side scripts and databases in order to prevent and detect website hijacking and unauthorized access to stored data. To achieve the required security, *SafeWS* utilizes a combination of lightweight cryptographic integrity and encryption tools, software engineering techniques, and security data management principles. We also describe our implementation of *SafeWS* and its evaluation. The performance analysis of our prototype shows the overhead introduced by security verification is small. *SafeWS* will allow business owners to significantly reduce the security risks and vulnerabilities of outsourcing their sensitive customer data to third-party providers.

## 1 Introduction

As e-commerce becomes more common on the Internet, an increasing number of small businesses (e.g., online stores) use hosting providers to open their doors to online customers. These small businesses put their trust in various hosting service providers for the benefits of higher availability, fast website access, round-the-clock support and a very low cost [6]. As a result, customer data, which may contain sensitive information (such as Social Security numbers or credit card information), is either stored by these third-party providers, or can be accessed from their servers.

In what follows, we will use the following terminology to distinguish the entities that concern us. *Service provider* refers to an organization and its employees which are in the business of leasing web service resources. *Web-server* refers to a machine and software running on it that is owned by a service provider<sup>1</sup>. A web-server provides website content to an end-user. *Website content* refers to HTML pages, scripts and any data in a database that is stored or served by the scripts on a given website. An *end-user* or a *customer* is a person that is interested in website content for a given website,

---

\* This work has been supported in part by NSF grant CNS-0831186 and the Rutgers University Computing Coordination Council Pervasive Computing Initiative Grant.

<sup>1</sup> Software refers to the entire non-hardware environment provided, including the operating system, web-server software (such as Apache) and PHP interpreter.

this person may disclose sensitive information to this website. A *website owner* is the person (or organization) that is in charge of website content and has interest in keeping end-user information safe. Web site owners lease web-servers from service providers in order to run their web site.

A website owner may purchase a (low cost) certificate to authenticate their establishment [5, 7]. When customers fill out HTML forms with their sensitive information on SSL protected websites, they are led to believe that with the padded lock icon and an authenticated signed certificate, the data that they are about to hand over is safe. However, SSL only protects the *end-to-end* security of the data from the customer's computer to the web-server [5], it bears no indication of the kind of protection that the data gets once it enters the domain of the service provider. As we will explain next, many website owners typically store the database credentials in clear-text. This information can be easily used to login to the website owners' databases and either retrieve or alter sensitive information of customers.

Many websites are driven by database systems, as databases are widely used to store customer data and product information and play a crucial and central role in modern e-commerce. Using a combination of server-side scripts<sup>2</sup> and database connections is a widely used approach in providing dynamic content to online users and in retrieving and storing customer data on databases. These scripts offer website owners a versatile interpreted language that can create complex web environments, with libraries offering connectivity to many other services such as databases. However, there has not been any framework provided within the web-server environment to allow for safe execution of server-side scripts. Server-side scripts are called by web server modules, which are initiated by end-user requests. Due to the interpretive nature of server-side scripts, and with the tools available today, it is impossible to obscure or hide sensitive program details from users with administrative access to the web-server machine. Furthermore, aside from being completely readable, these server-side scripts can be altered by privileged users without the owner's consent or knowledge.

Storing database passwords as *clear-text* in server-side scripts is the *de facto* practice for IT professionals world-wide. In an outsourced setting, this approach implies that it is extremely easy for malicious employees at the service provider organization to access the passwords of business owners and thus their customer data. Security breaches at providers, caused by outside adversaries, may also expose hosted sensitive information. However, neither the research nor the industrial community have been giving enough attention to protecting sensitive data from being used by unauthorized parties and untrusted service providers in the outsourced setting.

Server-side script creators have been aware of the clear-text credential issues [4, 19], but have been concentrating on the ability of an end-user to see them, in case the web-server software is badly configured or compromised. The service providers and their web servers have been assumed trustworthy, therefore no protection against insider threats with respect to server-side scripts has been provided. The existing solution to protect against credential disclosure to end-users is to create an external script containing the clear-text credentials, placing it outside of the document root, and including it by the called script. While this naive hiding strategy might protect against poor

---

<sup>2</sup> Such as PHP, ASP, PERL, Python, etc.

web-server software configuration, a tampered version of a web server, as well as people with access to the server's operating system, can read that file. Another approach is to include the credentials inside the web server configuration file [19], but this again does not protect against insider threats.

**Our Contributions.** In this paper, we take the first step to address the gap between the security needs in outsourced hosting services and the protection provided in the current practice. We consider both insider and outsider attacks in the third-party web hosting scenarios. We describe the threats and security vulnerabilities that exist in today's web environments. And finally we present *SafeWS*<sup>3</sup>, a system with a relatively low overhead to mitigate the threats of potential security breaches at service providers. *SafeWS* is a novel and modular solution that prevents and detects website hijacking and unauthorized access to stored data.

To achieve the required security, our solution utilizes a combination of lightweight cryptographic integrity and encryption tools, software engineering techniques, and security data management principles. Our solution is written in C/C++ and contains a component that resides between server-side scripts and database systems, as well as components that reside off the service-provider's server. We evaluated *SafeWS* with PHP scripts and the performance analysis shows the overhead introduced by *SafeWS* is small.

*SafeWS* allows business owners to significantly reduce the security risks and vulnerabilities of outsourcing their sensitive customer data to third-party providers. By deploying our solution, website owners can provide their customers with a robust and secure storage of their sensitive personal information.

*The main goal of this work is to improve the protection of outsourced sensitive data on untrusted web servers.* We believe that low-budgeted database driven websites that use shared-hosting have the greatest risk for unauthorized disclosure of information, and therefore designed *SafeWS* for their needs. Our framework enables website owners (e.g., small business owners) to automatically verify the integrity of service providers and their web servers. Most importantly, we efficiently and effectively prevent sensitive credentials, e.g., database locations, names, usernames and passwords, from being stored as clear-text at the service provider side. The highlights of our solution are shown below.

- Guarantees that only authorized webpages from the web server can access a database that stores end users or product information.
- Effectively hides database access credentials from web server administrators who can read any file on the system.
- Ensures the integrity of outsourced websites by detecting and monitoring suspicious environments and activities; provides website owner with notifications of suspicious activities.

*SafeWS* demonstrates a general security design principle for outsourced computation that is a contribution beyond the specific server-side script problem studied. Our work

---

<sup>3</sup> *SafeWS* stands for Safe Web Script.

can improve the security of any kind of database driven web server system in a third-party service provider setting.

**Scope of our work.** We decided to concentrate our work in the layer between server-side scripts and databases as that is the intersection where the end-user, the owner and the service provider meet. It allows us to verify the authenticity of owner-built scripts and the healthiness of the service provider environment, while also protecting end-user data from being maliciously used. At this layer we can also notify the owner of any foul play, thus minimizing the possible impact of an attack. Authorization and access control, if implemented within server side scripts, can also benefit from *SafeWS* as long as a database connection is used, though *SafeWS* is not meant to be an access control system. We also concentrated our efforts on preventing data theft and corruption, rather than handling denial of service attacks. Lastly, we decided to evaluate our system on a platform (LAMP<sup>4</sup>), that is widely used in the third-party hosting setting, is open-source and easily implemented, but due to the generality of our design, we believe our system to be valid for any server-side scripting language (*SafeWS* can also improve the security of non-scripting programs such as compiled/binary server-side programs).

**Organization of the paper.** The rest of the paper is organized as follows. Our adversarial model, security definitions, and assumptions are described in Section 2. The architecture and protocols are presented in Section 3. Our security analysis is discussed in Section 4. Performance evaluation of *SafeWS* is described in Section 5. Related work is given in Section 6. Finally, we describe future work and conclude in Section 7.

## 2 Definitions and Trust Models

In this section, we give the necessary definitions, trust model, adversary model, and security definitions used in our solution. First, let us briefly recapture our definitions on the types of entities introduced in Section 1.

- *Service provider* refers to an organization and its employees which are in the business of leasing web service resources.
- *Web-server* refers to a machine and software running on it that is owned by a service provider.
- An *end-user* or a *customer* is a person that is interested in website content for a given website.
- A *website owner* is the person (or organization) that is in charge of website content and has interest in keeping end-user information safe.

In reality, website owners typically want to provide a cost-effective solution to their customers. Most of them are unaware of or unable to comprehend the security requirements and guarantees in the outsourcing environments of service providers. A large number of website owners do not write server side scripts for data manipulation themselves. Instead, those functions are provided as part of the outsourcing service.

---

<sup>4</sup> Linux, Apache, MySQL and PHP.

Similarly, website customers are usually completely unaware of the business outsourcing agreements between website owners and service providers. Consequently, the end-users assume that their sensitive information is only released to the website owner, and no one else<sup>5</sup>.

**Trust Relationships.** Our trust model is simple and intuitive. The main interactions are between the website owner and the service provider. Website owners are not malicious. Web servers are not trusted by the website owners, and therefore, our solution is used by a website owner to verify the integrity of the outsourced environment. Website users trust that the website owner is ensuring the outsourced web server is not compromised.

**Adversarial Model in SafeWS.** Instead of assuming abstract adversaries, we strive to give a concrete and comprehensive categorization of types of attackers, from both inside or outside the service provider organization. Such a practical analysis is both crucial and fundamental to the security of proposed solutions. Because of the specific application scenario studied, we are able to describe a concrete adversarial model.

We divided the security threats on a hosted web server into different levels based on the position and experience of the possible attacker. While we would want to believe most people would not violate the trust put in them by their employer, all it takes is one person with a different set of motives. Every given level is assumed to encompass the abilities of the previous level. Thus, a novice hacker may do anything that an administrator could do.

We believe that some of these security threats can be reduced by making them more difficult to accomplish, as well as providing our own threat of recognizing an attack when it happens, and notifying the owners.

- *Nosy Administrator.* Any server administrator with super-user access can scan the directory tree for clear-text server-side script files. Upon finding those files, this person can then read the database credentials, and learn the names of the columns and tables where sensitive information is stored. This person can then decide to enter the provided database and look at the information stored there. While this person might be acting out of curiosity, the outcome is that an unauthorized person was able to view secret data.
- *Disgruntled Employee.* A disgruntled employee, or specifically a disgruntled employee with system access, may maliciously obtain information from hosted client websites in the same way the Nosy Administrator would, but for different reasons. This person can actually cause financial harm by disclosing information to outsiders, or using it for personal gain.
- *Novice Hacker.* In addition to all the threats defined above, a novice hacker may attempt to hijack a website by changing the server-side script files that obtain web-user information, or by adding new script files. A novice hacker may also try to replace the web-server executable with a malicious one.
- *Advanced Hacker.* An advanced hacker may analyze traffic in and out of the system, change the kernel, scan the memory, and reverse engineer any program.

---

<sup>5</sup> For server authentication, users can rely on their web browser indications, such as the lock icon, an https address and a valid certificate.

**Definitions of Security.** We formalize our security goals in three requirements, namely, *secrecy of database credentials*, *provenance authentication for database access*, and *integrity of outsourced environments*.

The *secrecy of database credentials* is defined as that the credentials (e.g., passwords) required to access the website owner's database need to be confidential and cannot be learned by privileged users on the service provider's machine, who can read *any file* on the system. We give the web server administrators significant amount of power, which is necessary in this outsourced scenario. This requirement implies that the website owner's database needs to be maintained by a different provider from the web server provider. In practice, such a separation of duties principle (i.e., separating web server provider from database provider) is in general desirable to constrain and balance providers' privileges.

*Provenance authorization for database access* is defined as that only authorized web-pages on the web server can access a database that stores end user or product information. The *provenance of a database request* is the webpage that initiates the database connection. This requirement means that the provenance information associated with a database request must be recorded, submitted, and verified before a request can be satisfied.

*Integrity of outsourced environments* is defined as that any tampering with the web and computing environments, including parameters, software, and libraries, should be detected and the website owners' notified.

**Security Assumptions.** The owner of a website can obtain the SHA-1 hashes of non-hacked Apache executable and modules running on the web-server before the server is compromised. The owner of a website has a separate, non-compromised machine where he can store private keys, authenticate server-side script files, periodically activate configuration scripts and compile *SafeWS*. The machine running *SafeWS* may be compromised after *SafeWS* is already in place. The service provider may, from time to time, upgrade the software on the web server. It is up to the owner to keep up with these changes and reconfigure *SafeWS* accordingly (as automatically identifying whether a change of a library or executable is done maliciously or not is outside our scope). If multiple script files which require database access include each other, it is up to the owner to ensure each of them calls *SafeWS* as we do not want to introduce the overhead of nested source files and pre-compilation to our run-time environment. Embedding connection strings in compiled code is an approach that can provide added security, as opposed to just placing them in clear-text scripts, as some work needs to be done to decompile and evaluate the data. This is common practice for any compiled (non-script) DB accessing CGI<sup>6</sup> on a web server, but it is not flexible, nor secure enough. Recompile is needed whenever credentials change, as the environment is not authenticated and the credentials are revealed after one decompilation.

### 3 Architecture

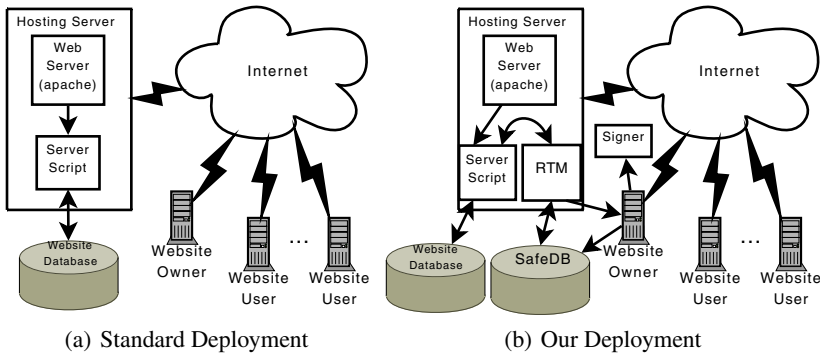
Intuitively, our solution provides the website owner a way to evaluate, assess, and authenticate the working environments of a web server hosted by a third-party service

---

<sup>6</sup> Common Gateway Interface.

provider. With our solution in place, a website owner (or his trusted technologically savvy agent) stores and encrypts part of the security information used for authentication on the third-party web server. If abnormal conditions are detected, our solution has the ability to automatically notify and alert the website owner and users to the well-being of the third-party server. Next, we will give detailed descriptions on the architecture, components, and procedures of our solution.

The design of our solution is divided into two parts. The first part includes all actions needed during run-time to ensure only authorized and authenticated scripts may access the database, and other attempts would cause notifications to be sent to the owner of the site. The second part is an offline process that happens mostly outside the server to ensure proper configuration of the solution<sup>7</sup>.



**Fig. 1.** Schematic drawings of the architecture of database-driven hosted website in standard deployment and with our solution

### 3.1 Security of Database Credentials

A big security vulnerability in outsourced environments is the existence of clear-text database credentials and connection strings inside server-side script files. Existing common practice is to place this information in another script file that resides outside of the document root and is included during run-time [4, 19]. While this simple approach might protect the included file from a poorly configured web-server, it does nothing to prevent a user (or a superuser) on the machine from reading it.

To solve the clear-text database credential problem, our approach is to hide the database credentials by encrypting them, and storing them in a separate database, which we call *SafeDB*. Because our solution includes access to *SafeDB*, we need to protect ourselves from having that access information freely available. We achieve this in two steps. First, we compile the module that accesses the database and derive its database access password from a signed SHA-1 hash of the module’s own executable. Second, this module checks if it was run by an authorized script file on a trusted web-server.

<sup>7</sup> The results of this process are the inputs for the run-time module and so may reside on the web-server.

Note that the location of this database may vary. Locating it locally with the web server may reduce the system's security, as a superuser may connect to and alter *SafeDB*. Locating it off the web server improves the security guarantees, while a distributed deployment may reduce the reliability and stability of the service. We further evaluate and analyze the performance in Section 5.

### 3.2 Key Generation and Solution Setup

During compile time, two sets of 2048-bit RSA keys are generated for the *Signer module* and *Run-Time Module (RTM)*, which are described in the next section. The keys are placed in header files included by the *Signer* and *RTM*, respectively. The SHA-1 hashes of a valid web-server executable and related modules are compiled into the *RTM* as well as the owner's e-mail address. Once the package is compiled, one can begin incorporating our solution into the website's script files. A part of an unchanged script file (PHP) is shown in Listing 1, where the IP address, database user, database password and the name of the database are all in clear-text. To demonstrate the ease of using our solution, we show how to convert the legacy code in Listing 1 to safer code in Listing 2.

**Listing 1.** PHP Script connecting to a Database

```
<?php
...
$db = mysql_connect(192.168.0.100, 'my_usr', 'PWord');
mysql_select_db('my_store_db');
...
?>
```

**Listing 2.** PHP Script connecting to a database using SafeWS. With SafeWS, sensitive database information is not exposed.

```
<?php
...
$info = safe_exec('/home/u1/bin/rtm', 'tag_f11');
list($db_host, $db_name, $db_user, $db_pass) =
    split(':', $info, 4);
mysql_connect($db_host, $db_user, $db_pass);
mysql_select_db($db_name);
...
?>
```

### 3.3 Run-Time Module and Signer Module

The Run-Time Module (*RTM*) is the most crucial component in our solution. It is called within an authorized script file using *safe\_exec()* (described below). When *RTM* loads,



it calculates the SHA-1 hashes of its own executable, the script file that called it, as well as the executable of the web server (i. e., the calling process) and its relevant modules. When executing, *RTM* is supplied with a *tag* parameter. This piece of information allows one script to request many different sets of Database credentials from *RTM*. *RTM* then checks that it was called by a valid web server by comparing SHA-1 hashes.

*RTM* then attempts to access *SafeDB*. If the module was tampered with, it will not be able to derive the correct database password from the SHA-1 hash of its own executable.

Upon connecting to the database, *RTM* looks for a record that matches the SHA-1 hash of the script filename (that called *RTM*) with the supplied *tag*. Using its private key, *RTM* decrypts one of the fields and verifies the signature of the *Signer* module, which is described next. If the hash of the script file is verified, then it is authentic, and the database connection parameters as well as credentials are returned to the calling script file.

The *Signer* module contains its own private key, as well as the *RTM*'s public key. This module resides on the website owner's machine, which is assumed to be safe. Whenever a new script file is ready to be put on the website, the owner converts it to be compatible with *SafeWS*.

The website owner uses the *Signer* to sign the SHA-1 hash of a script file that will be installed on the server, and encrypts the result with the *RTM*'s public key. The *Signer* also associates a tag with that script file and with the set of credentials given by the owner.

The output of the *Signer* is a *cryptographic SQL file* that includes the following information: a SHA-1 hash of the script full path name with the tag, credentials and database connection parameters encrypted with the public key of the *RTM* and the SHA-1 hash of the script file signed using the *Signer*'s private key and encrypted by the *RTM*'s public key. This SQL file is transferred to the server containing the *SafeWS* database and is executed there.

### 3.4 PHP Limitation and *safe\_exec()*

As we chose to evaluate our prototype with PHP scripts, we found a serious security limitation that affects PHP security and the security of PHP-based web hosting in general. In existing PHP execution environments, it is impossible to learn or verify the provenance (i.e., origin) of the caller.

PHP offers the following methods to execute non-PHP programs: *exec()*, *passthru()*, *proc\_open()*, *popen()*, *shell\_exec()* and *system()*. The process that is executed as a result of these calls does not know where the call originated (i.e. from which PHP file). Moreover, the parameters of the web-server session are not provided either. While it is possible to use *proc\_open()* and pass environment variables to a new process, this would undermine the information integrity, as a hacker might try to pass bogus parameters to bypass our protection. This security limitation may or may not affect other scripting languages.

To solve these problems, we had to add another module to *SafeWS*'s architecture. We created a new run-time PHP module called *safe\_exec()*. The main advantage of *safe\_exec()* is that it is able to pass web-session information as well as run-time environment information, including the calling PHP file, into the process that it executes.

*safe\_exec()* is integrated with our solution through *RTM*. This module needs to be installed on the service provider’s machine, and added to the list of files *RTM* authenticates. A similar module may be needed for other scripting languages which do not pass credible execution and web environment information to executed binaries. CGI (Common Gateway Interface) binaries executed directly by Apache do receive the required information, and thus can pass it through *execle()*.

### 3.5 SafeWS Run-Time Protocol

Once *SafeWS* is deployed and configured, it will be invoked by the web-server when a participating script file is processed. *SafeWS*’s protocol is illustrated in Figure 2 and is described as follows.

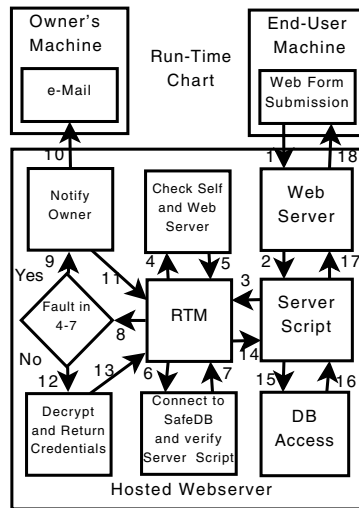


Fig. 2. SafeWS architecture and run-time information flow

- End-users submit an HTML form on their browser in Step 1, using either HTTP GET or HTTP POST.
- In Step 2, the hosted web-server executes the script that handles the data. The web-server then passes the information from the end-user to the script, through either environment variables or as the script’s standard input. After initial processing, the script needs to set the database connection parameters, in order to process the end-user’s request.
- The script calls *RTM* using *safe\_exec* in Step 3. Please refer to Listing 2 for an example of such a call.
- Once *SafeWS*’s *RTM* module is started, it computes the SHA-1 hash of its caller’s executable file (e.g. *apache*) in Step 4 and 5. It computes its own SHA-1 hash, which is then signed with *RTM*’s private key to produce the password for *SafeDB*.

- In Step 6 and 7, *RTM* attempts to access *SafeDB* with the credentials it was compiled with, as well as the password that it computed. *RTM* computes the SHA-1 hash of the caller PHP script, as well as a digest of the script's location and tag. The digest is used as a key to find the row in *SafeDB* that corresponds to the calling script. Upon selecting the relevant information from *SafeDB*, *RTM* attempts to verify the signed SHA-1 hash of the script. Namely, *RTM* (1) verifies the web-server executable and modules, (2) connects to *SafeDB*, (3) locates the correct record for the script, and (4) verifies the signature on its hash.
- Any failure in the above verification procedures results in *RTM* notifying the website owner of the security concern, as well as not returning the requested information to the calling script. This failure in turn would cause the script's subsequent connection to the database to fail, as shown in Step 9 through 11.
- In Step 12, upon a successful verification of the location and authenticity of the calling script, *RTM* decrypts the remainder of *SafeDB*'s record using its private key and obtains the address of the website's database server, as well as the database name, username and password required to access it. This information is then returned to the calling script in Step 13 and 14. The script then parses the data and connects to the website's database in Step 15 and 16, and can then complete its task.

In Figure 2 for the sake of description simplicity, we show *SafeDB* and the protected website's database all on the same local host as the web server. In practice, as we mentioned in Section 2, the database systems should preferably reside and be maintained at a different location from the local hosted web server, in order to reduce the security risk. Our protocol description and implementation can be directly used to accommodate such a distributed deployment of *SafeWS*.

### 3.6 Protecting against Advanced Hackers

As we stated in Section 2, advanced hackers may reverse engineer any program. This means that *RTM* would be vulnerable to attacks. If attackers reverse engineer *RTM* they could find the keys stored in it as well as the connection information to *SafeDB*. However, due to the hurdles we built into *SafeWS*, this process might take a considerable amount of time. First, *RTM* would have to be decompiled, the code (which may or may not resemble the original code) must be analyzed. The database credentials are made by using the SHA-1 hash of the *RTM* executable file signed by the private key stored in *RTM*. The attacker would need to build a program to use the extracted keys and obtain and sign the SHA-1 hash of *RTM*. In addition, the attackers must refrain from running or misusing *RTM* as the owners would be notified. Since we assume that *RTM* may be hacked, we can add another layer of security. By having a periodical job (such as a cron job) that would generate new keys, recompile both *RTM* and *Signer*, change the database credentials and distribute the new *RTM* to the server, we can reduce the probability of a successful attack. This periodical script would run on the trusted owner's machine, and would perform its duties every  $X - 1$  seconds where  $X$  is the minimum number of seconds that an advanced hacker would take to obtain the *SafeDB* credentials.

### 3.7 RTM Design Aspects

*RTM* is executed each time a script with database access is run by the web server. Although this is sub-optimal performance wise, the security aspects were more important. By letting *RTM* be resident in memory, we could implement caching of database credentials, as well as avoid re-examining the web server executable and its modules (provided we can guarantee that the process is the same). This would improve the performance of *SafeWS* considerably, however, there are a couple of caveats to this approach which made us choose the other design. First, the system is designed for shared hosting environments, which normally do not allow their customers (website owners) to have resident services running on the machine. Second, the lack of a direct execution relationship between a script and *RTM* would reduce the knowledge the system gives *RTM* about the caller, as well as complicate calling *RTM* inside the scripts.

## 4 Discussion

In this section, we analyze the security properties and discuss practical considerations associated with deploying *SafeWS*. *SafeWS* satisfies the security requirements that are defined in Section 2 including the secrecy of database credentials, provenance authentication for database access, and integrity of outsourced environments, which are explained in detail next.

*SafeWS* guarantees that only authorized webpages from the web server can access a database that stores end user or product information. This property is achieved by storing the properties and environments of authorized webpages in *SafeDB*, which can only be accessed by *RTM* with the proper privately generated password. These operations correspond to Step 6 and 7 in Figure 2. In addition, our basic script run-time module *safe\_exec()* ensures that only authentic information about script environments is passed to *RTM* for the verification purpose, and spoofing attacks (e.g., lying about an IP address or location) can be identified.

*SafeWS* effectively hides database access credentials from web server administrators who are allowed to read any file on the system. As stated earlier in Section 3, while using *SafeWS*, the website owner should separate the customer data from the running environment, i.e., scripts and data should reside on different servers. Sensitive data should be kept encrypted in a database, and the decrypting web server should be different than the encrypting one. With this separation of duties, even if attackers obtained the database credentials where the sensitive data is stored, they will not have a way to decrypt the encrypted customer data.

*SafeWS* ensures the integrity of outsourced websites by detecting and monitoring suspicious environments and activities and provides website owners with notifications of suspicious activities. The verification is realized mainly by our Run-Time Module in *SafeWS*. *RTM* leverages *safe\_exec()*'s ability to pass web-session information as well as run-time environment information into the process that it executes. In the *SafeWS* run-time protocol, *RTM* checks the integrity of the web-server executable and verifies the signature on the hash of the invoked script against the correct record in *SafeDB* that can only be accessed by *RTM*.

Typically when server-side scripts are first developed, they are changed often due to programming errors or inappropriate specifications. But most of such scripts reach stable states and are rarely changed afterwards. Therefore, updates in *SafeWS* caused by script changes do not cause much communication and computation overheads. We give a more thorough evaluation and discussion on the performance of *SafeWS* in Section 5.

Note that in-memory code mutation or memory scanning are types of attacks that may find the private key or clear-text passwords [12]. Our current *SafeWS* design can withstand advanced reverse engineering attacks against *RTM* with the help of a periodically running script, as described in Section 3.6. However, we believe that these types of attacks would take significant efforts. Due to the fact that the *Signer* module is safely located on the website owner's private machine, even if the *RTM*'s private key is recovered, the attacker will not be able to sign altered or new script files without the owner's key, and thus will be unable to hijack the website without being detected.

## 5 Evaluation

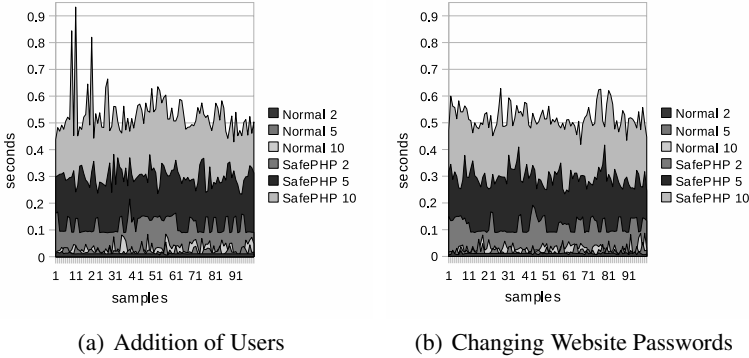
Our goals were to see whether *SafeWS* was a viable solution for small and medium third-party hosted web-sites. We decided to check the impact on both end-users and web server machines. As end-users today expect fast response times, we wanted to achieve sub-second end-to-end times and low server impact.

### 5.1 Experiment Setup

We used two servers for testing *SafeWS*. Our web server machine is an Intel dual core (2\*1.6Ghz) with 1MB cache and 2GB RAM. This machine runs Linux kernel version 2.6.23.17-88 SMP, Apache 2.2.8 and MySQL 5.0.45. Our client emulator and remote database machine is an Intel dual core (2\*2.8Ghz), with 1MB cache and 2GB RAM running Linux version 2.6.27.9 SMP and MySQL 5.0.67. We picked two common website procedures that may allow access to sensitive information. We wrote PHP scripts that store and update a database with this information. We measured the end-to-end performance of those functions, then converted the PHP scripts to work with *SafeWS* and re-measured. Each of the following experiments were conducted using both local (on the web server machine) as well as remote (on the client emulator machine) databases.

**Addition of Users.** We created a PHP script that processes an 'HTTP GET' form which creates a new web-site user. There were seven pieces of information stored for each user: First name, Last name, Address, City, Zip code, e-mail address and password. We then ran a program that produced random values for these fields and ran a multi-threaded program that generated varying amounts of concurrent sessions. We measured each session from start to finish.

**Changing Website Passwords.** We created another PHP script that handles changes to a website-user's password. The fields provided were the users e-mail address, the old password, and the new password. Using the stored generated data from the previous procedure, we generated new passwords and ran our session generator with varying amounts of concurrent sessions and measured their end-to-end performance.



**Fig. 3.** Concurrency vs. Response Times with and without SafeWS. Normal refers to non-SafeWS measurements, numbers refer to concurrent sessions.

## 5.2 Experiment Results

We were able to achieve a sustainable peak performance of over 72,000 user addition and password changing requests per hour. Although the web server running the RTM had a load average of 12, the average end-user experience was under 0.5 seconds (from browser connection to the web server until the response was fully available). Using slightly less concurrency (5 simultaneous requests at all times) we achieved a sustainable average of 0.3 seconds end-user response time and a server load average of 1.4. This translates to 57,600 requests per hour or as much as 1.38 million requests per day (for uniform distribution of visits). Our measurements showed no significant performance difference between local vs. remote database use. We believe this is due to us using a machine on the same LAN, as well as a reduction in resource consumption. Although the performance overhead of *SafeWS* is significantly over the average running time of the scripts that do not use *SafeWS* (refer to Figure 3), the end-to-end performance is still sub-second, and can be improved further by optimizing the web server and database server software.

## 6 Related Work

With the increasing development of IT outsourcing, a substantial amount of research work has been done on how to verify outsourced data and computation [2, 3, 9–11, 13–17]. Merkle hash trees have been used extensively for authentication of data elements [14]. Aggregate signatures are another approach for data authentication, where each data tuple is signed by the data owner [17]. Most recently, the privacy issue in verifying queries was first addressed by in [18] which gave an elegant solution using hashing for proving the completeness of selection queries without revealing neighboring entries.

Database-as-a-service (DAS) model [10, 11, 16, 16] is an instantiation of the computing model involving trusted clients, who store their data at an untrusted server that is administrated by a service provider. The challenge in DAS is to make it impossible

for the service provider to correctly interpret the data. The data is owned by clients. The clients only have limited computational power and storage, and they rely on the server for the mass computational power and storage. Hacigümüs, Iyer, and Mehrotra [11] addressed the execution of aggregate queries over encrypted data using homomorphic encryption scheme. Mykletun and Tsudik [16] proposed an alternative approach where the data owner pre-computes and encrypts the aggregate results and stores them at the service provider. This approach avoids the use of homomorphic encryption, which was found to have a security flaw when used for DAS [16]. Our model is different from DAS, and is suitable for a more general security setting, as the data does not have to originate from the client.

Efforts to discern the trustworthiness of a server (and in some cases alert web users to untrusted servers) utilizing hardware such as the Trusted Platform Module (TPM) [8] and using commitments and attestations [1, 13] and their combinations [20] have been made. However, these solutions do not protect against obtaining database credentials from a text file, and also require specialized hardware and kernel modification on the web-server side, as well as software on the client side, and trusted authorities to provide verification. In comparison, *SafeWS* is easy to adopt and more efficient.

## 7 Conclusions and Future Work

Outsourced information is as safe as the security provided by the server storing it. In order to improve the security of outsourced websites, we presented *SafeWS* in this paper. *SafeWS* is a protocol encompassing a distributed architecture that provides a robust layer of security between web server-side scripts and databases, while notifying site owners of anomalous run-time behavior. We gave the security models and definitions associated with *SafeWS* in the outsourced web service scenario. We implemented *SafeWS* system in C/C++ and performed extensive experimental evaluation on the performance and robustness of the system. Our results showed that the security overhead introduced by *SafeWS* is low at the web server side even when the number of users is large.

For future work, we plan to leverage the infrastructure provided by *SafeWS* to extend the protection to cross-site scripting (XSS). One promising approach is to add the identifiers of allowed referer pages into the *SafeWS* database, the same way as we retrieve, store, and verify this information from the web server. We also plan to further improve the performance and robustness of the *SafeWS* implementation.

## Dedication

The authors would like to dedicate this paper in memory of Denitsa Tilkidjieva. A dear friend and a bright third-year Ph.D. student at the Computer Science department in Rutgers. She passed away January 22nd, 2009 and will always be missed.

## References

1. Arbaugh, W.A., Farber, D.J., Smith, J.M.: A secure and reliable bootstrap architecture. In: Proceedings of the 1997 IEEE Symposium on Security and Privacy, pp. 65–71. IEEE Computer Society, Los Alamitos (1997)

2. Bertino, E., Ooi, B.C., Yang, Y., Deng, R.H.: Privacy and ownership preserving of outsourced medical data. In: Proceedings of the 21st International Conference on Data Engineering (ICDE), pp. 521–532 (2005)
3. Devanbu, P., Gertz, M., Martel, C., Stubblebine, S.: Authentic third-party data publication. *Journal of Computer Security* 11(3) (2003)
4. Dickinson, P.: Top 7 PHP Security Blunders (December 2005), <http://www.sitepoint.com/article/php-security-blunders/>
5. Dierks, T., Rescorla, E.: The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246 (Proposed Standard) (August 2008), <http://www.ietf.org/rfc/rfc5246.txt>
6. Foster, I., Kesselman, C., Nick, J.M., Tuecke, S.: Grid services for distributed system integration. *Computer* 35(6), 37–46 (2002)
7. GoDaddy.com. Why You Need An SSL Certificate, <https://www.godaddy.com/gdshop/pdf/SSLMarketingGuideGodaddy.pdf>
8. Trusted Computing Group. TCG 1.2 specifications, <https://www.trustedcomputinggroup.org/>
9. Hacigümüs, H., Iyer, B., Li, C., Mehrotra, S.: Executing SQL over encrypted data in the database-service provider model. In: Proceedings of ACM SIGMOD Conference on Management of Data, pp. 216–227. ACM Press, New York (2002)
10. Hacigümüs, H.B., Iyer, H.B., Mehrotra, S.: Providing database as a service. In: Proceedings of International Conference on Data Engineering (ICDE) (March 2002)
11. Hacigümüs, H., Iyer, B., Mehrotra, S.: Efficient execution of aggregation queries over encrypted relational databases. In: Lee, Y., Li, J., Whang, K.-Y., Lee, D. (eds.) DASFAA 2004. LNCS, vol. 2973, pp. 125–136. Springer, Heidelberg (2004)
12. Halderman, J.A., Schoen, S.D., Heninger, N., Clarkson, W., Paul, W., Calandrino, J.A., Feldman, A.J., Appelbaum, J., Felten, E.W.: Lest we remember: Cold boot attacks on encryption keys. In: van Oorschot, P.C. (ed.) USENIX Security Symposium, pp. 45–60. USENIX Association (2008)
13. Lampson, B., Burrows, M., Wobber, E.: Authentication in distributed systems: Theory and practice. *ACM Transactions on Computer Systems* 10, 265–310 (1992)
14. Merkle, R.: Protocols for public key cryptosystems. In: Proceedings of the 1980 Symposium on Security and Privacy, pp. 122–133. IEEE Computer Society Press, Los Alamitos (1980)
15. Mykletun, E., Narasimha, M., Tsudik, G.: Authentication and integrity in outsourced databases. In: Proceedings of Symposium on Network and Distributed Systems Security (NDSS) (February 2004)
16. Mykletun, E., Tsudik, G.: Aggregation queries in the database-as-a-service model. In: Damiani, E., Liu, P. (eds.) Data and Applications Security 2006. LNCS, vol. 4127, pp. 89–103. Springer, Heidelberg (2006)
17. Narasimha, M., Tsudik, G.: Authentication of outsourced databases using signature aggregation and chaining. In: Li Lee, M., Tan, K.-L., Wuwongse, V. (eds.) DASFAA 2006. LNCS, vol. 3882, pp. 420–436. Springer, Heidelberg (2006)
18. Pang, H., Jain, A., Ramamritham, K., Tan, K.-L.: Verifying completeness of relational query results in data publishing. In: Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD), pp. 407–418 (2005)
19. Shiflett, C.: Security corner: Shared hosting. *php—architect* 3(3) (March 2004), <http://shiflett.org/articles/shared-hosting>
20. Xu, G., Borcea, C., Iftode, L.: Satem: Trusted service code execution across transactions. In: IEEE Symposium on Reliable Distributed Systems, pp. 321–336 (2006)



# Inferring Trust Based on Similarity with TILLIT

Mozhgan Tavakolifard<sup>1</sup>, Peter Herrmann<sup>2</sup>, and Svein J. Knapskog<sup>1</sup>

<sup>1</sup> Centre for Quantifiable Quality of Service in Communication Systems (Q2S), Norwegian University of Science and Technology (NTNU), Trondheim, Norway  
mozhgan@q2s.ntnu.no, knapskog@q2s.ntnu.no

<sup>2</sup> Department of Telematics (ITEM), Norwegian University of Science and Technology (NTNU), Trondheim, Norway  
herrmann@item.ntnu.no

**Abstract.** A network of people having established trust relations and a model for propagation of related trust scores are fundamental building blocks in many of today's most successful e-commerce and recommendation systems. However, the web of trust is often too sparse to predict trust values between non-familiar people with high accuracy. Trust inferences are transitive associations among users in the context of an underlying social network and may provide additional information to alleviate the consequences of the sparsity and possible cold-start problems. Such approaches are helpful, provided that a complete trust path exists between the two users. An alternative approach to the problem is advocated in this paper. Based on collaborative filtering one can exploit the like-mindedness resp. similarity of individuals to infer trust to yet unknown parties which increases the trust relations in the web. For instance, if one knows that with respect to a specific property, two parties are trusted alike by a large number of different trusters, one can assume that they are similar. Thus, if one has a certain degree of trust to the one party, one can safely assume a very similar trustworthiness of the other one. In an attempt to provide high quality recommendations and proper initial trust values even when no complete trust propagation path or user profile exists, we propose TILLIT — a model based on combination of trust inferences and user similarity. The similarity is derived from the structure of the trust graph and users' trust behavior as opposed to other collaborative-filtering based approaches which use ratings of items or user's profile. We describe an algorithm realizing the approach based on a combination of trust inferences and user similarity, and validate the algorithm using a real large-scale data-set.

## 1 Introduction

Many online communities are only successful if sufficient mutual trust between their members exists. Users want to know whom to trust and how much to trust in the competence and benevolence of other community members in a specific application domain. The process of building trust is hereby performed in two different ways. First, one can establish trust (or distrust) by gaining direct experience with another party. Of course, every positive event increases the assumed trustworthiness of the trustee while every negative one reduces it. Second, one can gain trust based on recommendations of third

parties. If, e.g., Alice has high trust in Bob’s ability to assess the trustworthiness of other people, Bob has similar trust in Claire’s recommendations, and Claire considers David trustworthy based on her personal experience with him, then Alice gains also trust in David even if she has no or very limited knowledge of him at all. This form of propagated trust is called trust transitivity.

Based on the two forms of trust, a so-called web of trust between community members is created which is often used in recommender systems helping users of e-commerce applications to get an idea about the trustworthiness of their mostly personally unknown cooperation partners. Unfortunately, however, these webs of trust are often too sparse to be helpful in practice since — at least in large online communities — a user has experience with only a very small fraction of the other community members. Thus, very often there will be no trust relation to an intended new partner of an e-commerce transaction at all [14].

As a model to increase the number of trust relations, we propose the method TILLIT<sup>1</sup> (Trust Inference Links based on Like-minded Interaction Transitions). It enables to derive trust not only from direct experience and by transitive propagation but also from the similarity between users and vice versa. In particular, two users are considered similar if they either built akin trust relations to other users or if they are trusted very similarly by others. This can be used to propagate already known trust to new trust relations encompassing people similar to those of the yet known relationships. Thus, the web of trust can be augmented significantly.

In our model, we measure similarity based on the existing web of trust in a community using an iterative fixed-point algorithm on node-pair graphs introduced later in this paper. As a method to describe the values of trust as well as its propagation we apply the TNA-SL model [12] which is based on the Subjective Logic [10]. Our approach, however, would also work with other methods like [1, 7].

In comparison with other approaches based on similarity, our work has the following differences:

- It intends to alleviate the sparsity problem in the web of trust matrix itself instead of the matrix of users rating items in the system. Since users have usually few items rated in common, the classic recommender system techniques are often ineffective and are not able to compute a user similarity weight for many of the users. Instead, exploiting the web of trust, it is possible to propagate trust better and to infer additional trust information about other users.
- It calculates the similarity from the structure of the web of trust and trust relations (the trust graph structure and trust values) instead of user-item ratings.
- It proposes methods to convert trust values to similarity measures and vice versa based on the TNA-SL model.

We conducted experiments on a large real dataset showing how our proposed solution increases the coverage (number of trust relations that are predictable) while not reducing the accuracy (the error of predictions). This is especially true for users who have provided few ratings.

---

<sup>1</sup> “Tillit” is the Norwegian word for trust.

The rest of this paper is organized as follows: In section 2, we briefly explain the TNA-SL model as the background of our work. Our proposed model for trust inference is described in section 3. Next in section 4, we present the evaluation plan and results. Section 5 provides an overview of the related research. Finally, discussion and conclusion are given in section 6.

## 2 Trust Network Analysis with Subjective Logic

Our model is mainly based on TNA-SL [12], a model for trust network analysis. TNA-SL uses the Subjective Logic [10] which enables to represent a specific belief calculus. There trust is expressed by a belief metric called opinion. An opinion is denoted by  $\omega_B^A = (b, d, u, a)$  expressing the belief of a relying party  $A$  in the trustworthiness of another party  $B$ . The parameters  $b$  and  $d$  represent the belief resp. disbelief in  $B$ 's trustworthiness while  $u$  expresses the uncertainty of  $A$  about to trust  $B$  or not. The three parameters are all probability values between 0 and 1 and fulfill the constraint  $b + d + u = 1$ . The parameter  $a$  is called the base rate, and determines how uncertainty shall contribute to the opinion's probability expectation value which is calculated as  $E(\omega_x^A) = b + au$ . The opinion space can be mapped into the interior of an equal-sided triangle, where, the three parameters  $b$ ,  $d$ , and  $u$  determine the position of the point in the triangle representing the opinion. Fig 1 illustrates an example where the opinion is  $\omega_x = (0.7, 0.1, 0.2, 0.5)$ .

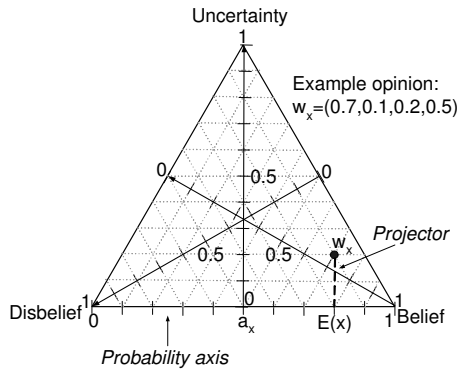


Fig. 1. Opinion triangle with an example opinion [10]

Based on TNA-SL, there are two different types of trust relations: *functional trust* (FT) and *referral trust* (RT). The former concerns  $A$ 's direct trust in  $B$  performing a specific task; the latter concerns  $A$ 's trust in  $B$  giving a recommendation about someone else doing a task or in other words is the trust in the ability to refer to a third party. As mentioned in the introduction, the simplest form of trust inference is trust transitivity which is widely discussed in literature [3, 8, 19, 24, 27]. That is, if  $A$  trusts  $B$  who trusts  $C$ , then  $A$  will also trusts  $C$ . A valid transitive trust path requires that the last edge in the

path represents functional trust and that all other edges in the path represents referral trust. Referral trust transitivity and parallel combination of trust paths are expressed as part of TNA-SL model (figure 2) [12].

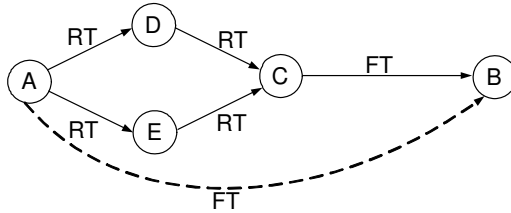


Fig. 2. Referral trust transitivity and parallel combination of trust paths

The discounting operator ( $\otimes$ ) [11] is used to derive trust from transitive trust paths, and the consensus operator ( $\oplus$ ) allows to combine parallel transitive trust paths. The trust network in figure 2 can then be expressed as

$$FT_B^A = ((RT_D^A \otimes RT_C^D) \oplus (RT_E^A \otimes RT_C^E)) \otimes FT_B^C$$

While we consider TNA-SL and the Subjective Logic as a suitable fundament for our similarity model, it can be, as already mentioned, adapted to all trust management models enabling to combine referral and functional trust (e.g., [1, 7]).

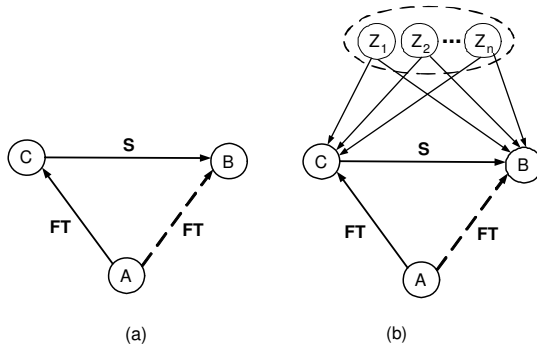
### 3 The Proposed Model

Our model for the estimation how much trust  $A$  can place in  $B$  considers not only direct experience and recommendations but also similarities between agents with respect of trusting other agents or being trusted by other parties. The two kinds of similarities between trusters resp. trustees can be gradually expressed by triples very similar to the first three operands of the opinion quadruples such that we can use the consensus operator of the subjective logic for the trust value computation.

#### 3.1 Similar Trustees

If  $A$  has functional trust in  $C$  who is similar to  $B$  (they are *similar trustees*), then  $A$  can infer its functional trust to  $B$  ([3], see figure 3(a)). Two trustees are similar if they are both similarly trusted by other agents  $Z_1, Z_2, \dots, Z_n$  (figure 3(b)). This is an extension of TNA-SL in which it is not possible to infer any trust value of  $A$  towards  $B$  in a trust network.

Similarly to Jøsang’s way to define opinions, we use triples to describe similarity which enables us to consider uncertainty. In particular, the degree of similarity depends on the number  $n$  of agents  $Z_1, Z_2, \dots, Z_n$  used for the computation reflecting that we are more certain about the similarity of two parties if they are trusted by a significant large number of other agents in an akin way.



**Fig. 3.** (a) Similar trustees (b) Similarly trusted

**Definition 1.** The similarity opinion  $S_B^C$  from  $C$  towards  $B$  is the triple<sup>[2]</sup> (similarity, non-similarity, uncertainty). If  $C = B$ , the similarity opinion is defined to be  $(1, 0, 0)$ . Otherwise, it is calculated based on the measure  $sim_{te}(C, B)$  of similarity between the two trustees  $C$  and  $B$  which is introduced in subsection 3.3.

$$S_B^C = \left( \frac{n \cdot sim_{te}(C, B)}{c + n}, \frac{n \cdot (1 - sim_{te}(C, B))}{c + n}, \frac{c}{c + n} \right) \quad (1)$$

$c$  is a constant determining how fast uncertainty is replaced by assurance. As higher its value is, as more agents are needed to reduce the uncertainty value in favor of the similarity and non-similarity values. The similarity opinion fulfills the constraints that the sum of all three values is equal to 1.

Our similarity opinion is a special form of referral trust. It reflects that the akin trust evaluations of  $B$  and  $C$  by several other trusters are a kind of recommendation by these agents to  $A$  to treat  $B$  and  $C$  similarly. Thus, we see the discounting operator  $\otimes$  as the correct mechanism to combine the similarity opinion between  $B$  and  $C$  with the functional trust of  $A$  in  $C$  in order to infer the functional trust of  $A$  in  $B$ :

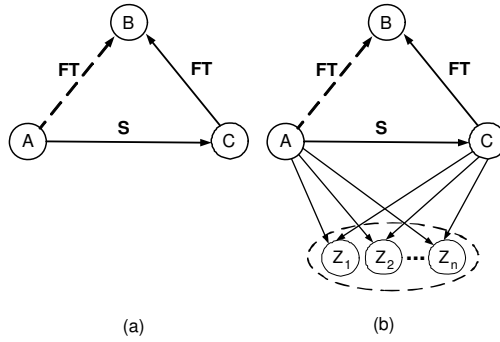
$$FT_B^A = S_B^C \otimes FT_C^A \quad (2)$$

As higher the similarity between  $B$  and  $C$  is, as closer the trust of  $A$  to  $B$  will equal to that between  $A$  and  $C$ . As lower this similarity is, as more uncertain  $A$  will be about whether to trust  $B$  or not.

### 3.2 Similar Trusters

If  $C$  has functional trust to  $B$  and  $A$  is similar to  $C$  (they are *similar trustees*), then  $A$  can also infer functional trust towards  $B$  ([13], see figure 4(a)). We call  $C$  and  $A$  similar trustees if they have alike trust in several other agents  $Z_1, Z_2, \dots, Z_n$ . In this case, if  $C$  has functional trust to a new agent  $B$ , then  $A$  can infer a functional trust to  $B$  (figure 4(b)). Again using TNA-SL alone, there is no way to infer a new trust value.

<sup>2</sup> This metric is inferred from a metric for the trust value computation [13] by Jøsang and Knapskog.



**Fig. 4.** (a) Similar trusters (b) Similarly trusting

Like (II), the similarity opinion  $S_C^A$  from  $A$  to  $C$  is calculated using the measure of similarity  $sim_{tr}(C,A)$  between trusters which is also introduced in subsection 3.3:

$$S_C^A = \left( \frac{n \cdot sim_{tr}(C,A)}{c+n}, \frac{n \cdot (1 - sim_{tr}(C,A))}{c+n}, \frac{c}{c+n} \right) \tag{3}$$

This similarity opinion is discounted by the functional trust  $FT_B^C$  from  $C$  to  $B$  to form the new trust value.

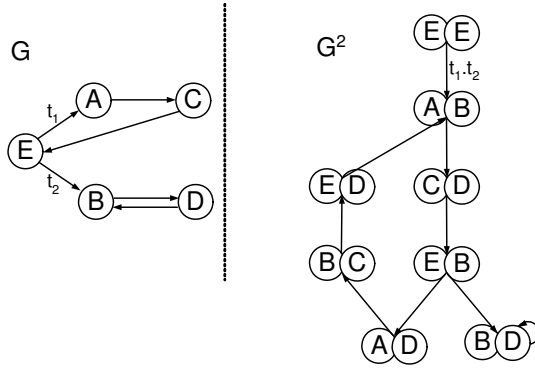
$$FT_B^A = S_C^A \otimes FT_B^C \tag{4}$$

### 3.3 Similarity Calculation

In order to measure similarities, we model trusters, trustees, and trust relationships as a graph with nodes representing trusters and trustees and edges representing trust relationships. The intuition behind our algorithm is that, *similar* trustees are related to *similar* trusters. More precisely, trusters  $A$  and  $B$  are similar if they are related to trustees  $C$  and  $D$ , respectively, and  $C$  and  $D$  are themselves similar. The base case is that each node is similar to itself. If we call this graph  $G$ , then we can form a node-pair graph  $G^2$  in which each node represents an ordered pair of nodes of  $G$  as depicted in figure 5. A node  $(A,B)$  of  $G^2$  points to a node  $(C,D)$  if, in  $G$ ,  $A$  points to  $C$  and  $B$  points to  $D$ . Similarity scores are symmetric, so for clarity we draw  $(A,B)$  and  $(B,A)$  as a single node  $A,B$  (with the union of their associated edges) [9].

We propose an iterative fixed-point algorithm on  $G^2$  to compute similarity scores<sup>3</sup> for node-pairs in  $G^2$ . The similarity score for a node  $v$  of  $G^2$  gives a measure of similarity between the two nodes of  $G$  represented by  $v$ . Scores can be thought of as flowing from a node to its neighbors. Each iteration propagates scores one step forward along the direction of the edges, until the system stabilizes (i.e., scores converge). Since nodes of  $G^2$  represents pairs in  $G$ , similarity is propagated from pair to pair. Under this computation, two trustees are similar if they are trusted by similar trusters.

<sup>3</sup> An alternative approach to measure this similarity is to model an agent’s mental structure as an ontology and using various methods proposed in our previous work [25, 26].



**Fig. 5.** Similarity measurement

For each iteration  $k$ , iterative similarity functions  $sim_{te,k}(*,*)$  for trustees and  $sim_{tr,k}(*,*)$  for trusters are introduced. The iterative computation is started with  $sim_{0,*}(*,*)$  defined as

$$sim_{0,*}(A,B) = \begin{cases} 1, & \text{if } A = B \\ 0, & \text{if } A \neq B \end{cases} \quad (5)$$

On the  $(k+1)$ -th iteration,  $sim_{*,k+1}(*,*)$  is defined in special cases as

$$\begin{aligned} sim_{*,k+1}(A,B) &= 1, & \text{if } A = B \\ sim_{te,k+1}(A,B) &= 0, & \text{if } I(A) = \emptyset \text{ or } I(B) = \emptyset \\ sim_{tr,k+1}(A,B) &= 0, & \text{if } O(A) = \emptyset \text{ or } O(B) = \emptyset \end{aligned} \quad (6)$$

$I(A)$  is the set of in-neighbors of  $A$  while  $O(A)$  specifies the set of  $A$ 's out-neighbors. Individual in-neighbors are denoted as  $I_i(A)$ , for  $1 \leq i \leq |I(A)|$ , and individual out-neighbors are denoted as  $O_i(A)$ , for  $1 \leq i \leq |O(A)|$ .  $sim_{te,k+1}(*,*)$  is computed from  $sim_{tr,k}(*,*)$  in the general case as follows:

$$sim_{te,k+1}(A,B) = \frac{\sum_{i=1}^n \sum_{j=i}^n sim_{tr,k}(I_i(A), I_j(B)) \cdot (1 - \text{distance}(I_i(A), I_j(B), A, B))}{\sum_{i=1}^n \sum_{j=i}^n sim_{tr,k}(I_i(A), I_j(B))} \quad (7)$$

and  $sim_{tr,k+1}(*,*)$  is computed from  $sim_{te,k}(*,*)$  in the general case as:

$$sim_{tr,k+1}(A,B) = \frac{\sum_{i=1}^n \sum_{j=i}^n sim_{te,k}(O_i(A), O_j(B)) \cdot (1 - \text{distance}(A, B, O_i(A), O_j(B)))}{\sum_{i=1}^n \sum_{j=i}^n sim_{te,k}(O_i(A), O_j(B))} \quad (8)$$

Formulas (7) and (8) are alternately computed in iterations until the resulting similarity values  $sim_{tr}$  and  $sim_{te}$  converge. The corresponding algorithm is sketched as the procedure *CalculateSimilarity* in algorithm [11](#).

The *distance* function is used to compare trust relations.  $distance(A, B, C, D)$  expresses the difference between the trust from  $A, B$  to  $C, D$ . It averages the Euclidean distances between the trust values of  $A$  and  $C$  resp.  $B$  and  $D$  on the opinion triangle (see figure 1):

$$\begin{aligned} distance(A, A, C, D) &= \sqrt{(b_{AC} + \frac{1}{2}u_{AC} - b_{AD} - \frac{1}{2}u_{AD})^2 + \frac{3}{4}(u_{AC} - u_{AD})^2} \\ distance(A, B, C, C) &= \sqrt{(b_{AC} + \frac{1}{2}u_{AC} - b_{BC} - \frac{1}{2}u_{BC})^2 + \frac{3}{4}(u_{AC} - u_{BC})^2} \\ distance(A, B, C, D) &= \begin{cases} \frac{1}{2}(\sqrt{(b_{AC} + \frac{1}{2}u_{AC} - b_{BD} - \frac{1}{2}u_{BD})^2 + \frac{3}{4}(u_{AC} - u_{BD})^2} \\ + \sqrt{(b_{AD} + \frac{1}{2}u_{AD} - b_{BC} - \frac{1}{2}u_{BC})^2 + \frac{3}{4}(u_{AD} - u_{BC})^2}) \end{cases} \end{aligned} \quad (9)$$

For the sake of simplicity, all base rate values ( $a_{AD}, a_{AC}, a_{BD}, a_{BC}$ ) are assumed to be  $\frac{1}{2}$ . The factor  $\frac{3}{2}$  is used for the vertical axis to adapt the measures. Otherwise, the opinion triangle would be compressed and the distance between the points  $(0,1,0)$  and  $(0,0,1)$  would not be equal to one. Figure 6 illustrates the *distance* function graphically.

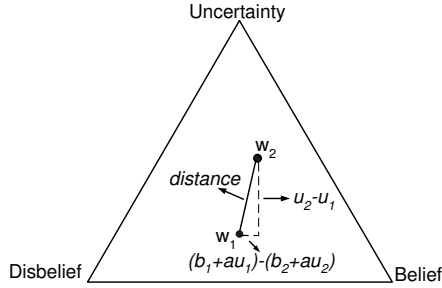


Fig. 6. The distance between opinions

## 4 Evaluation

We chose a publicly available dataset taken from a real system known as Advogato [2]. Advogato (<http://advogato.org>) is an online community site dedicated to free software development. On Advogato a user can certify another user as “Master”, “Journeyer”, “Apprentice” or “Observer”, based on the perceived level of involvement in the free software community. The Advogato social network is an example of a real-world, directed, weighted, large social network. There are indeed other web communities using the same software powering Advogato.org and they also have reached similar trust levels and use the same certifications system, but we do not use them for our analysis in this paper, mainly because:

- Our model is based on user-user trust matrix and not the user-item rating matrix.
- They are much smaller than the Advogato dataset.

### 4.1 Dataset

Precise rules for giving out trust statements are specified on the Advogato site. *Masters* are supposed to be principal authors of an “important” free software project, excellent



programmers who work full time on free software. *Journeyers* contribute significantly, but not necessarily full-time. *Apprentices* contribute in some way, but are still acquiring the skills needed to make more significant contributions. *Observers* are users without trust certification, and this is the default. It is also the level at which a user certifies another user to remove previously expressed trust certifications.

The Advogato dataset is a directed, weighted graph with 11934 nodes and 57610 trust relations. There are 18053 Master judgments, 23091 for Journeyer, 10708 for Apprentice and 5758 for Observers. Figure 7 illustrates the allocation of ratings that correspond to each user. In our tests, we apply our model to 3 different datasets and the results are averaged. Each 3000 users built a trust graph of approximately 4000 relations.

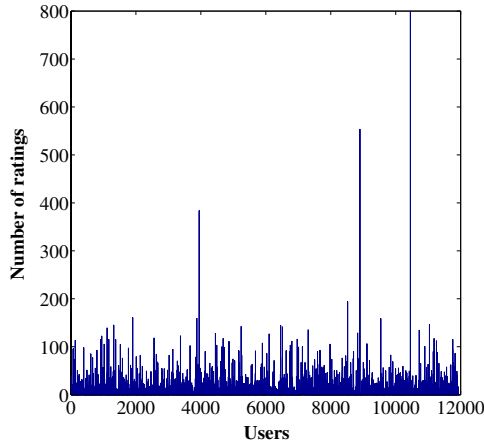


Fig. 7. Users’ rating activity

For the purpose of this paper, we consider these certifications as trust statements. Trust statements are directed and not necessarily symmetric. By aggregating the trust statements expressed by all the members of the community it is possible to build the entire trust network. A trust network is hence a directed, weighted graph. Arbitrarily, we map the textual labels Observer, Apprentice, Journeyer and Master respectively to rating values 0, 1, 2, 3. which have to be yet converted to subjective logic opinions. In general, with n-level rating values (in our case  $n = 3$ ) in which the number of ratings of level  $i$  is described by function  $f(i)$ , we can use the following conversion method in which  $c$  is a constant:

$$b = \frac{\sum_{i=1}^n i \cdot f(i)}{c + n \cdot \sum_{i=0}^n f(i)}, \quad d = \frac{\sum_{i=0}^{n-1} (n - i) \cdot f(i)}{c + n \cdot \sum_{i=0}^n f(i)}, \quad u = \frac{c}{c + n \cdot \sum_{i=0}^n f(i)} \quad (10)$$

In this formula, the highest rating value 3 is mapped to three positive valuations, while 2 corresponds to two positive valuations and a negative one, etc.

**Algorithm 1.** EVALUATION(*users, trust\_graph*)

---

```

procedure CALCULATESIMILARITY(users, trust_graph)
  repeat
    for each  $i, j \in \text{users}$ 
      do if  $i = j$ 
        then  $\text{similarity\_matrix}[i, j] \leftarrow (1, 0, 0)$ 
      else if  $i < j$ 
        then  $\text{neighbors} \leftarrow$  common in-neighbors of  $i$  and  $j$ 
        comment: similarity of trustees
        else  $\text{neighbors} \leftarrow$  common out-neighbors of  $i$  and  $j$ 
        comment: similarity of trusters
      else if  $\text{number\_of\_neighbors} == 0$ 
        then  $\text{sim} \leftarrow 0$ 
        else  $\text{sim} \leftarrow$  GETSIMILARITY( $\text{neighbors}$ )
        comment: According to (7) and (8)
       $\text{similarity\_matrix}[i, j] \leftarrow$  GETOPINION( $\text{sim}, \text{number\_of\_neighbors}$ )
      comment: According to (11)
    until converge
  return ( $\text{similarity\_matrix}$ )

procedure PREDICTTRUSTEDGE( $(i, j), \text{trust\_graph}$ )
   $\text{opinion} \leftarrow (0, 0, 1)$ 
  for each  $k \in \text{users} - \{i, j\}$ 
    do {
       $\text{similarity\_trustee}(k, j) \leftarrow \text{similarity\_matrix}[\min(k, j), \max(k, j)]$ 
       $\text{similarity\_truster}(i, k) \leftarrow \text{similarity\_matrix}[\max(i, k), \min(i, k)]$ 
       $\text{predicted\_opinion\_te} \leftarrow \text{trust\_opinion}(i, k) \otimes \text{similarity\_trustee}(k, j)$ 
       $\text{predicted\_opinion\_tr} \leftarrow \text{trust\_opinion}(k, j) \otimes \text{similarity\_truster}(i, k)$ 
       $\text{opinion} \leftarrow (\text{opinion} \oplus \text{predicted\_opinion\_te} \oplus \text{predicted\_opinion\_tr})$ 
    }
  return ( $\text{opinion}$ )

procedure DOEVALUATION( $\text{trust\_graph}, \text{predicted\_trust\_graph}$ )
   $\text{coverage} \leftarrow$  number of predicted edges in  $\text{predicted\_trust\_graph}$ 
   $\text{fcpe} \leftarrow$  fraction of correctly predicted edges
   $\text{mae} \leftarrow$  mean absolute error of predicted values
   $\text{rmse} \leftarrow$  root mean squared error of predicted values
  output ( $\text{coverage}, \text{fcpe}, \text{mae}, \text{rmse}$ )

main
  global  $\text{similarity\_matrix} \leftarrow$  CALCULATESIMILARITY( $\text{users}, \text{trust\_graph}$ )
  for each  $\text{edge} \in \text{trust\_graph}$ 
    do {
       $\text{predicted\_edge} \leftarrow$  PREDICTTRUSTEDGE( $\text{edge}, \text{trust\_graph} - \text{edge}$ )
       $\text{predicted\_trust\_graph} \leftarrow \text{predicted\_trust\_graph} \cup \text{predicted\_edge}$ 
    }
  DOEVALUATION( $\text{trust\_graph}, \text{predicted\_trust\_graph}$ )

```

---

### 4.2 Plan

We use the leave-one-out technique [4] (a machine learning evaluation technique) to show the performance of our approach. Leave one out involves hiding one trust edge and then trying to predict it. The predicted trust edge is then compared with the real edge (using the distance function) and the difference is the prediction error. This procedure is repeated for all edges in the trust graph. The real and the predicted values are then compared in several ways: the coverage, which refers to the fraction of edges for which, after being hidden, the algorithm is able to produce a predicted edge, FCPE which is the fraction of correctly predicted edges, MAE (mean absolute error) which is average of the prediction error over all edges, and RMSE (root mean squared error) which is the root mean of the average of the squared prediction error. RMSE tends to emphasize large errors.

The evaluation can be described in pseudo-code as in algorithm 1. First, the similarity matrix is calculated by calling the procedure *CalculateSimilarity* from the main procedure. Since similarity is symmetric, the similarity of trustees is stored in the lower triangle of the similarity matrix and the similarity of trusters in the upper triangle. Next, for each edge in the real trust graph, an equivalent trust edge is calculated by calling procedure *PredictTrustEdge*. This procedure takes the real trust graph without that edge as an input. The predicted edges form the predicted trust graph. Finally, the real and predicted trust graph are compared according to the four metrics (coverage, FCPE, MAE, and RSME) by calling procedure *DoEvaluation*.

### 4.3 Results Summary

Figure 8 depicts the similarity measures among the first 150 users. For each two users, their similarity as trustees is in the lower triangle of the similarity matrix and their similarity as trusters is in the upper triangle of the similarity matrix.

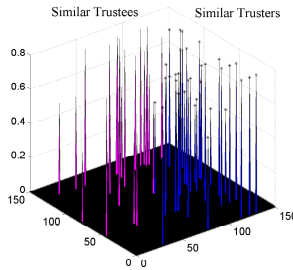


Fig. 8. Similarity measures among the first 150 users

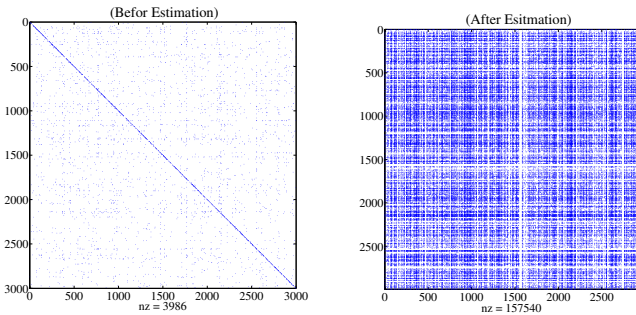
In table 1 we present the final results of the evaluation. We start by commenting the column “coverage”. The coverage becomes an important issue on a very sparse dataset that contains a large portion of cold start users since many trust values become hardly

predictable [17]. Our baseline is a method called “Random” which randomly generates trust edges. Results ( $coverage \approx 0.6$ ) indicate that our model is able to predicate approximately one edge from each two existing edges. The second important result is the fraction of correctly predicted edges (FCPE) which is 0.8. It shows that from each 10 predicted edge 8 edges are predicted correctly. Further, prediction errors (MAE and RMSE) computed are small in comparison with the Random method ( $MAE \approx 0.14$  &  $RMSE \approx 0.18$ ).

**Table 1.** Final evaluation results

Metric	Dataset1	Dataset2	Dataset3	Average	Random
Coverage	0.5783	0.5678	0.6520	0.5994	1
FCPE	0.8169	0.8299	0.8227	0.8232	0.3068
MAE	0.1389	0.1427	0.1409	0.1408	0.4570
RMSE	0.1823	0.1828	0.1864	0.1838	0.5036

Figure 9 shows the sparsity of the trust graph before and after prediction for the first dataset. The sparseness has been decreased significantly. All-in-all, the results of the evaluation lead to the expectation that the method TILLIT will increase the coverage of trust relationships significantly, and that the accuracy of the predicted additional will be fairly high as well.



**Fig. 9.** Sparsity of the trust graph before and after prediction for the first dataset

## 5 Related Research

Most popular approaches proposed to deal with the sparsity problem include dimensionality reduction of the user-item matrix, application of associative retrieval techniques in the bipartite graph of items and users, item-based similarity instead of user-based similarity, and content-boosted collaborative filtering (see [21]). The dimensionality reduction approach addresses the sparsity problem by removing unrepresentative or insignificant users or items so as to condense the user-item matrix. We briefly explain

those which are based on similarity measurement and thus more closely resemble our work. These approaches can be categorized in two groups: rating-based similarity and profile-based similarity.

In [22, 23], Pitsilis and Marshall explain how similarity can benefit from special characteristics of trust such as the ability to propagate along chains of trusted users; in this way similarity can support transitivity. In their model they use ordinary measures of similarity taken from collaborative filtering to form the potential trust between the users which would be propagated in a similar way to the word-of-mouth scheme through a trust graph. Finally, by transforming the value back into similarity measure terms, it could be made appropriate for use in collaborative filtering algorithms. More specifically, for each pair of users they first calculate how similar they are, applying Pearsons correlation coefficient formula over the user-item ratings, and then they calculate the indirect trust between them. Next, this trust value is converted to a similarity metric using their formula.

Massa et al. present in [16, 18] evidence that, by incorporating trust, recommender systems can be more effective than systems based on traditional techniques like collaborative filtering. They show how the similarity measure, on average, is computable only against a very small portion of the user base and is, in most cases, a noisy and unreliable value because computed on few items rated in common by two users. Instead, trust-aware techniques can produce a trust score for a very high number of other users; the trust score of a user estimates the relevance of that users' preferences. In this paper, similarity is measured using Pearsons correlation coefficient on user-item ratings.

A number of techniques for performing collaborative filtering from the point of view of a trust-management problem are outlined in [15]. In this work authors propose a variation of k-nearest neighbor collaborative filtering algorithm for trusted k-nearest recommenders. This algorithm allows users to learn who and how much to trust one another by evaluating the utility of the rating information they receive. They mainly address the problem of learning how much to trust rating information that is received from other users in a recommender system.

A model for computing trust-based reputation for communities of strangers is proposed in [5]. The model uses the concept of knots, which are sets of members having high levels of trust in each other. Different knots typically represent different view points and preferences. The assumption underlying this knot-aware reputation model is that use of relatively small, but carefully selected, subsets of the overall community's reputation data yields better results than those represented by the full dataset.

In [20], O'Donovan and Smyth argue that profile similarity on its own may not be sufficient, that other factors might also have an important role to play. Specifically they introduce the notion of trust in reference to the degree to which one might trust a specific profile when it comes to make a specific rating prediction. They develop two different trust models, one that operates at level of the profile and one at level of the items within a profile. In both of these models trust is estimated by monitoring the accuracy of a profile at making predictions over an extended period of time. Trust then is the percentage of correct predictions that a profile has made in general (profile-level trust) or with respect to a particular item (item-level trust).

In [28], the authors experimentally prove that there exists a significant correlation between the trust expressed by the users and their profile similarity based on the recommendations they made in the system. This correlation is further studied as survey-based experiments in [6].

In this paper we provide an alternative approach to deal with the sparsity problem. We measure similarity based on the users' trust relationships, i.e. trust graph structure and trust values (in contrast to the other approaches which have used user-item ratings or profile similarity), and propose novel formulas to convert it to subjective logic opinions. The consideration of these similarities leads to extra information accessible for trust inferences.

## 6 Discussion and Conclusion

In order to overcome sparseness of the web of trust, we consider users' similarity as a factor to derive trust connectivity and trust values. The main idea is that we account two persons similar if either a fair number of others have akin trust in them or if they themselves trust several other people alike. In the first case, every person who has trust in one of them can infer similar trust to the other one, at least as an estimated starting value. In the second case, a person may infer the trust value of a third party from other trusters similar to her.

We consider a similarity-based recommendation system for singers and songs as a good application example for our model. Normally, in systems like iTunes only the most popular songs or other songs of artists, of whom one already has bought songs, are advertised without any guarantee that one likes these songs as well. Using our approach, it is possible to find other customers who have an akin taste about music as the customer Alice reading the advertisements. Songs rated positively by these customers but not bought yet by Alice can be advertised to her since she will like them probably as well. This will make Alice more receptive to the advertisements.

In the future, we aim to evaluate the accuracy of a whole recommender system that employs our proposed model. Furthermore, we assess the possibility of modeling some of other trust propagation methods using our approach. An example is transposition

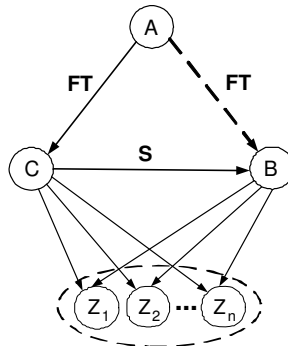


Fig. 10. Coupling: a trust propagation method

resp. reciprocity [8] assuming that  $A$ 's trust in  $B$  causes  $B$  to develop also some level of trust towards  $A$ . Another propagation method is Coupling, in which  $A$ 's trust in  $C$  propagates to  $B$  because  $C$  and  $B$  trust people in common [8]. This propagation rule is depicted in figure 10. According to this rule we can use the similarity between trusters to propagate the trust in one trustee to another.

Moreover, one can use similarity in a complete different way. Trust is very specific and nobody trusting Bob as a good car mechanic will automatically trust him also in undertaking heart surgeries. But probably, he will be capable in repairing motorcycles. Thus, there is a large similarity between the domains of repairing cars and motorcycles but a very low one between both of these and medical surgery. We think to use trust relations in one domain to infer ones in similar domains and consider ontologies describing the degrees of similarity between the domains as a useful means. All-in-all, we are convinced, that the various forms of similarity are good vehicles to tackle the major problem of too sparse webs of trust in online communities.

## References

1. Abdul-Rahman, A., Hailes, S.: Supporting trust in virtual communities. In: Proceedings of the 33rd Hawaii International Conference, Maui, Hawaii, vol. 6. IEEE Computer Society Press, Los Alamitos (2000)
2. [http://www.trustlet.org/wiki/Advogato\\_dataset](http://www.trustlet.org/wiki/Advogato_dataset)
3. Ding, L., Kolari, P., Ganjugunte, S., Finin, T., Joshi, A.: Modeling and Evaluating Trust Network Inference. Technical report, Maryland Univ. Baltimore Dept. of Computer Science and Electrical Engineering (2005)
4. Fukunaga, K., Hummels, D.M.: Leave-one-out procedures for nonparametric error estimates. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 11(4), 421–423 (1989)
5. Gal-Oz, N., Gudes, E., Hendler, D.: A Robust and Knot-Aware Trust-Based Reputation Model. In: Proceedings of IFIPTM 2008 - Joint iTrust and PST Conferences on Privacy, Trust Management and Security, pp. 167–182. Springer, Boston (2008)
6. Golbeck, J.: Trust and nuanced profile similarity in online social networks. *Journal of Artificial Intelligence Research* (2006)
7. Grandison, T., Sloman, M.: Specifying and analysing trust for internet applications. In: Proceedings of the 2nd IFIP Conference on E-Commerce, E-Business & E-Government (I3E), Lisbon, pp. 145–157. Kluwer Academic Publisher, Dordrecht (2002)
8. Guha, R., Kumar, R., Raghavan, P., Tomkins, A.: Propagation of trust and distrust. In: Proceedings of the 13th international conference on World Wide Web, pp. 403–412. ACM Press, New York (2004)
9. Jeh, G., Widom, J.: SimRank: a measure of structural-context similarity. In: Proceedings of the eighth ACM SIGKDD international conference on Knowledge Discovery and Data Mining, pp. 538–543. ACM Press, New York (2002)
10. Jøsang, A.: A Logic for Uncertain Probabilities. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 9(3), 279–311 (2001)
11. Jøsang, A.: The consensus operator for combining beliefs. *Artificial Intelligence* 141(1-2), 157–170 (2002)
12. Jøsang, A., Hayward, R., Pope, S.: Trust network analysis with subjective logic. In: Proceedings of the 29th Australasian Computer Science Conference, vol. 48, pp. 85–94. Australian Computer Society (2006)

13. Jøsang, A., Knapskog, S.J.: A metric for trusted systems. In: Proceedings of the 21st National Security Conference, NSA (1998)
14. Kim, Y.A., Le, M.T., Lauw, H.W., Lim, E.P., Liu, H., Srivastava, J.: Building a web of trust without explicit trust ratings. In: IEEE 24th International Conference on Data Engineering Workshop, ICDEW 2008, pp. 531–536 (2008)
15. Lathia, N., Hailes, S., Capra, L.: Trust-Based Collaborative Filtering. In: Proceedings of IFIPTM 2008 - Joint iTrust and PST Conferences on Privacy, Trust Management and Security, pp. 119–134. Springer, Boston (2008)
16. Massa, P., Avesani, P.: Trust-Aware Collaborative Filtering for Recommender Systems. In: Meersman, R., Tari, Z. (eds.) OTM 2004. LNCS, vol. 3290, pp. 492–508. Springer, Heidelberg (2004)
17. Massa, P., Avesani, P.: Trust-aware recommender systems. In: Proceedings of the 2007 ACM conference on Recommender systems, pp. 17–24. ACM Press, New York (2007)
18. Massa, P., Bhattacharjee, B.: Using Trust in Recommender Systems: An Experimental Analysis. In: Jensen, C., Poslad, S., Dimitrakos, T. (eds.) iTrust 2004. LNCS, vol. 2995, pp. 221–235. Springer, Heidelberg (2004)
19. Morselli, R., Bhattacharjee, B., Katz, J., Marsh, M.: Exploiting approximate transitivity of trust. In: Fourth International Conference on Broadband Communications, Networks and Systems, BROADNETS 2007, pp. 515–524 (2007)
20. O'Donovan, J., Smyth, B.: Trust in recommender systems. In: Proceedings of the 10th international conference on Intelligent User Interfaces, pp. 167–174. ACM, New York (2005)
21. Papagelis, M., Plexousakis, D., Kutsuras, T.: Alleviating the sparsity problem of collaborative filtering using trust inferences. In: Herrmann, P., Issarny, V., Shiu, S.C.K. (eds.) iTrust 2005. LNCS, vol. 3477, pp. 224–239. Springer, Heidelberg (2005)
22. Pitsilis, G., Marshall, L.: Model of Trust Derivation from Evidence for Use in Recommendation Systems. Technical report, Newcastle University, School of Computing Science (2004)
23. Pitsilis, G., Marshall, L.F.: Modeling Trust for Recommender Systems using Similarity Metrics. In: Proceedings of IFIPTM 2008 - Joint iTrust and PST Conferences on Privacy, Trust Management and Security, pp. 103–118. Springer, Boston (2008)
24. Quercia, D., Hailes, S., Capra, L.: Lightweight Distributed Trust Propagation. In: Seventh IEEE International Conference on Data Mining, ICDM 2007, pp. 282–291 (2007)
25. Tavakolifard, M., Knapskog, S., Herrmann, P.: Cross-Situation Trust Reasoning. In: Proceedings of The Workshop on Web Personalization, Reputation and Recommender Systems (WPRRS 2008). IEEE Computer Society Press, Los Alamitos (2008)
26. Tavakolifard, M., Knapskog, S., Herrmann, P.: Trust Transferability Among Similar Contexts. In: Proceedings of The 4th ACM International Workshop on QoS and Security for Wireless and Mobile Networks (Q2SWinet 2008). ACM, New York (2008)
27. Yang, Y., Brown, L., Lewis, E., Newmarch, J.: W3 Trust Model: Evaluating Trust and Transitivity of Trust of Online Services. In: International Conference on Internet Computing, pp. 354–362 (2002)
28. Ziegler, C.N., Golbeck, J.: Investigating interactions of trust and interest similarity. *Decision Support Systems* 43(2), 460–475 (2007)



# Analogical Trust Reasoning

Mozhgan Tavakolifard<sup>1</sup>, Peter Herrmann<sup>2</sup>, and Pinar Öztürk<sup>3</sup>

<sup>1</sup> Centre for Quantifiable Quality of Service in Communication Systems (Q2S), Norwegian University of Science and Technology (NTNU), Trondheim, Norway

mozhgan@q2s.ntnu.no

<sup>2</sup> Department of Telematics (ITEM), Norwegian University of Science and Technology (NTNU), Trondheim, Norway

herrmann@item.ntnu.no

<sup>3</sup> Computer and Information Science Department (IDI), Norwegian University of Science and Technology (NTNU), Trondheim, Norway

pinar@idi.ntnu.no

**Abstract.** Trust is situation-specific and the trust judgment problem with which the trustor is confronted might be, in some ways, similar but not identical to some problems the trustor has previously encountered. The trustor then may draw information from these past experiences useful for the current situation. We present a knowledge-intensive and model-based case-based reasoning framework that supports the trustor to infer such information. The suggested method augments the typically sparse trust information by inferring the missing information from other situational conditions, and can better support situation-aware trust management. Our framework can be coupled with existing trust management models to make them situation-aware. It uses the underlying model of trust management to transfer trust information between situations. We validate the proposed framework for Subjective Logic trust management model and evaluate it by conducting experiments on a large real dataset.

## 1 Introduction

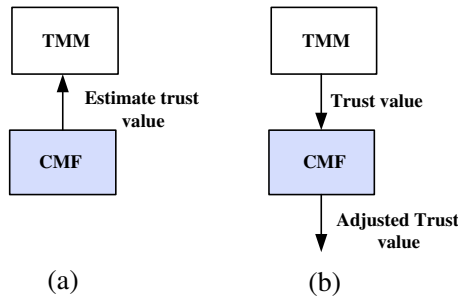
This paper presents a context management framework (CMF) that employs case-based reasoning [19] to analyze the correlation between trust information among various situations and help to bootstrap in unanticipated situations using trust information available from similar situations. The case-based reasoning technique is particularly useful for tasks that are experience-intensive, that involve plausible (i.e. not sound) reasoning and have incomplete rules to apply.

The fundamental principle of the case-based reasoning technique is similar to that of the human analogical reasoning process which employs solutions of past problems to solve current ones. The reasoning process is generally composed of three stages: remembering, reusing, and learning. Remembering is the case-retrieval process, which retrieves relevant and useful past cases. In the reusing step, the case-based reasoning system applies the cases that have been retrieved to find an effective solution to the current problem. Learning is the process of casebase enhancement. At the end of each problem-solving session the new case and problem-solving experiences incorporated into the casebase [15].

We present a universal mechanism (called CMF) that can be combined with existing trust management models (TMM) to extend their capabilities towards efficient modeling of the situation-aware trust by

- estimating the trust values based on similar situations, in unknown situations or for unknown trustees when there is no information available. Therefore, CMF can help TMM to bootstrap (Figure 1(a)).
- adjusting the output of TMM (trust value) based on the underlying situation, thus, providing situation-awareness for TMM (Figure 1(b)).

In our approach TMM is implemented using the Subjective Logic [12]. One of our main contributions is the extension of the Subjective Logic with a context-sensitive domain model.



**Fig. 1.** Scope and interconnection of context management framework (CMF) and trust management model (TMM). a) Estimation of the trust value in unknown situations. b) Adjustment of the output of TMM (trust value) based on the underlying situation.

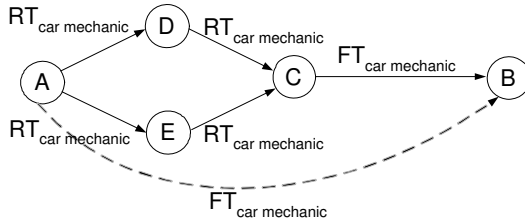
The rest of this paper is organized as follows: In section 2 we briefly explain the Subjective Logic as an example of the trust management model. Our proposed model for trust inference is described in section 3. Next in section 4 we present the evaluation plan and results. Section 5 provides an overview of the related research. Finally, conclusion and some ideas for future work are given in section 6.

## 2 Subjective Logic Trust Management Model

In this section, we briefly explain the Subjective Logic fundamentals and give reasons why it needs to be extended with a *situation* dimension. Subjective Logic [10] enables the representation of a specific belief calculus in which trust is expressed by a belief metric called opinion. An opinion is denoted by  $\omega_B^A = (b, d, u, a)$  expressing the belief of a relying party  $A$  in the trustworthiness of another party  $B$ . The parameters  $b$  and  $d$  represent the belief respectively. disbelief in  $B$ 's trustworthiness while  $u$  expresses the uncertainty in  $A$ 's trust in  $B$ . All the three parameters are probability values between 0 and 1, and fulfill the constraint  $b + d + u = 1$ . The parameter  $a$  is called the base rate and determines how uncertainty contributes to the opinion's probability expected

value which is calculated as  $E(\omega_x^A) = b + au$ . The opinion space can be mapped into the interior of an equal-sided triangle, where the three parameters  $b$ ,  $d$ , and  $u$  determine the position of the point in the triangle representing the opinion.

Based on the Subjective Logic, there are two different types of trust relations: *functional trust* ( $FT_B^A$ ) and *referral trust* ( $RT_B^A$ ). The former concerns  $A$ 's direct trust in  $B$  performing a specific task, while the latter concerns  $A$ 's trust in  $B$  giving a recommendation about someone else doing a task. In other words, it is the trust in the ability to refer to a suitable third party. The simplest form of trust inference is trust transitivity which is widely discussed in literature [4, 7, 23]. That is, if  $A$  trusts  $B$  who trusts  $C$ , then  $A$  will also trusts in  $C$ . A valid transitive trust path requires that the last edge in the path represents functional trust and that all other edges in the path represents referral trust. Referral trust transitivity and parallel combination of trust paths are expressed as part of the Subjective Logic model (figure 2) [12].



**Fig. 2.** Trust transitivity and parallel combination of trust paths. FT is functional trust and RT is referral trust.

The discounting operator ( $\otimes$ ) [11] is used to derive trust from transitive trust paths, and the consensus operator ( $\oplus$ ) allows to combine parallel transitive trust paths. The trust network in figure 2 can then be expressed as

$$FT_B^A = ((RT_D^A \otimes RT_C^D) \oplus (RT_E^A \otimes RT_C^E)) \otimes FT_B^C \tag{1}$$

There are two reasons for extension of the Subjective Logic with situation representation. First, It has been shown [3] that trust is not always transitive in real life. For example, the fact that  $A$  trusts  $B$  to fix her car and  $B$  trusts  $C$  to look after his child does not imply that  $A$  trusts  $C$  for fixing the car, or for looking after her child. However, under certain semantic constraints, trust can be transitive and a trust referral system can be used to derive transitive trust. The semantic constraint in the Subjective Logic is that the subject of trust should be the same along the entire path, for example all trust subjects should be “to be a good car mechanic” (figure 2) or “looking after her child”. On the other hand, this constraint is relaxed in our proposal by introducing the notion of situation. We suggest that trust situations along a transitive trust path can be different but similar to each other. For instance, trust situations can be “to be a good car mechanic” or “to be a good motor mechanic” (figure 3). In this way, we are able to use trust information from available similar situations (section 6 provides the details).

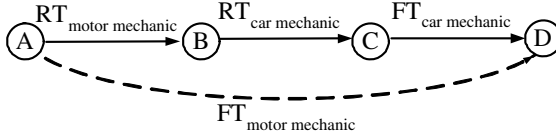


Fig. 3. Trust transferability among similar situations

Second, Jøsang introduces three different versions of the consensus operator (denoted by  $\oplus$ ,  $\oplus$ ,  $\oplus$  respectively) for fusion of independent, dependent, and partially dependent trust opinions [14]. If  $A$  and  $B$  have simultaneously observed the same event in the situation then their opinions are dependent. If  $A$  and  $B$  observed the same event during two partially overlapping situations then their opinions are partially dependent (e.g.  $A$  and  $B$  observed the same event of fire at the same time.  $A$  was in the place of fire, while  $B$  saw it on TV). Jøsang assumes that fraction of the overlapping observations is known and proposes formulas to estimate dependent and independent parts of the two observations to define the consensus operator of partially dependent opinions ( $\oplus$ ). We propose to calculate the fraction of overlapping observations as the similarity measure between the two situations.

### 3 The Proposed Framework

We consider two approaches for the inference task among situations: rule-based inference and similarity-based reasoning, depicted respectively as case-based reasoner (CBR) and rule-based reasoner (RBR) modules in figure 4. The former provides the

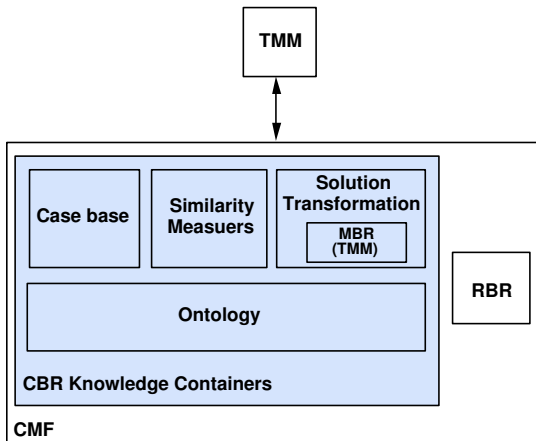


Fig. 4. Knowledge containers in case-based reasoner (CBR). TMM: trust management model, MBR: Model-based reasoner, RBR: rule-based reasoner, CMF: context management framework.

first role (Figure 1(a)), estimation of the trust value in unanticipated situations and the latter is responsible for the second role (Figure 1(b)) of CMF, adjustment of the trust values based on underlying situation. The gray box in figure 4 shows the focus of this paper.

### 3.1 Case-Based Reasoner Module

In the case-based reasoning approach, knowledge is distributed among the four knowledge containers: ontology, casebase, similarity measures, and solution transformation.

- *Ontology*: We represent the situations in the pertinent domain in form of an ontology. A situation consists of set of *contexts* which are captured as nodes of the ontology. Figure 5 depicts the ontology related to user-movie ratings. In this example, a situation has two main contexts: *User* and *Movie*. Demographic information for the users (age, occupation, sex, and zip code) are *local contexts* for the *User* context and movie genres are *local contexts* for the *Movie* context.

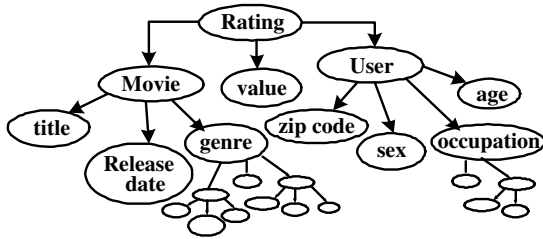
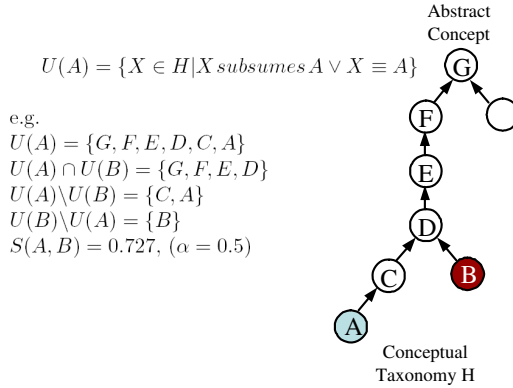


Fig. 5. The ontology example for user-movie ratings

- *Casebase*: The characterizations of the previous experiences and the recommendations (trust information including truster, trustee, trust value, and situation) are stored as elements of cases in the casebase. Cases are represented as attribute-value pairs.
- *Similarity*: The similarity between situations is a weighted sum of the similarity between their contexts. Similarity between contexts, in turn, are computed as the weighted sum of the similarity between the underlying local contexts. According to the Tversky's formula [30], the similarity between two concepts *A* and *B* can be determined in the following way:

$$S(A,B) = \frac{|U(A) \cap U(B)|}{|U(A) \cap U(B)| + \alpha |U(A) \setminus U(B)| + (1 - \alpha) |U(B) \setminus U(A)|} \quad (2)$$

$U(A)$  and  $U(B)$  are the sets of properties of concepts *A* and *B*, respectively. The function  $U$  takes into account the depth of compared concepts in the ontology hierarchy.  $\alpha$  is a value in the range  $[0, 0.5]$ . The value of 0 implies that the differences of *A* with respect to *B* are not sufficient to conclude that they are similar, and the value



**Fig. 6.** Relations taxonomy

of 0.5 means that the differences are necessary and sufficient to conclude such an assumption. Figure 6 illustrates an example of the similarity calculation.

In our approach, equation (2) is used to compare the attributes with each other, while the comparison between the values of an attribute is performed using the following general comparison guidelines:

- *Categorical*: values in the same category are similar (e.g., weather).
- *Continuous*: closer values are alike (e.g., time).
- *Hierarchical*: values in the same hierarchy are similar (e.g., location).

Attributes which do not have these characteristics may require a custom comparator to be defined for them.

- *Solution transformation*: The model-based reasoner (MBR) is responsible for adaptation or transformation of a solution (trust value) from previous experiences to the current problem of trust judgment. It uses TMM to estimate trust value for the current situation based on trust values of the similar situations (see figure 4). In section 3.2.1 we consider the Subjective Logic model as TMM and provide details for the solution transformation module.

### 3.2 Processes

CMF is generally composed of three processes: Remembering, Reusing, and Learning.

- *Remembering*: The query (the current trust assessment question) is compared to cases (past trust assessment experiences) in the casebase and N most similar cases are retrieved (N nearest neighbors). This process uses the ontology to measure the similarity between the query and each case in the casebase.
- *Reusing*: A trust value is predicted for the query using the solution transformation module.
- *Learning*: A new case is built from the query and the predicted value and is added to the casebase for future uses.

<sup>1</sup> In [27] we provide a comprehensive set of similarity measurement algorithms.

In following, we explain the details for solution transformation module considering the Subjective Logic as TMM.

### 3.2.1 Solution Transformation in Case of the Subjective Logic

We explain the functionality of the model-based reasoner through extension of the Subjective Logic model as TMM. If  $A$  has functional trust in  $B$  in situation  $C_1$ , then  $A$  can infer its functional trust to  $B$  in situation  $C_2$  which is a similar situation. For example, if  $A$  trusts  $B$  as a good car mechanic then  $A$  will probably trust  $B$  in repairing motorcycles since there is a large similarity between the domains of repairing cars and motorcycles.

Similarly to Jøsang's way to define opinions, we use triples to describe similarity which enables us to use the Subjective Logic operators.

**Definition 1.** *The similarity opinion  $S_{C_1}^{C_2}$  from  $C_1$  towards  $C_2$  is the triple  $(s, d, u)$  (similarity, non-similarity, uncertainty) and fulfills the constraints that the sum of all three values is equal to 1. If  $C_1 = C_2$ , the similarity opinion is defined to be  $(1, 0, 0)$ . Otherwise, it is calculated based on the measure of similarity ( $S(C_1, C_2)$ ) between the two situations  $C_1$  and  $C_2$  and the depth of concepts in the ontology (see (2)):*

$$S_{C_1}^{C_2} = \left( \frac{S(C_1, C_2) \cdot UN(C_1, C_2)}{k + UN(C_1, C_2)}, \frac{(1 - S(C_1, C_2)) \cdot UN(C_1, C_2)}{k + UN(C_1, C_2)}, \frac{k}{k + UN(C_1, C_2)} \right) \quad (3)$$

Here,  $k$  is a constant and  $UN(C_1, C_2) = |U(C_1) \cup U(C_2)|$  defining the number of properties in play at all. In general, the higher the similarity value is, the less uncertain we are, and the uncertainty will be lower as more details ( $UN(C_1, C_2)$ ) are available in comparison of the two situations  $C_1$  and  $C_2$ .

Our similarity opinion is a special form of referral trust. It reflects that the akin situations of  $C_1$  and  $C_2$  is a kind of recommendation (reminding) to  $A$  to treat in situations  $C_1$  and  $C_2$  similarly. Thus, we see the consensus operator  $\otimes$  as the correct mechanism to combine the similarity opinion between  $C_1$  and  $C_2$  with the functional trust of  $A$  in  $B$  in order to infer the functional trust of  $A$  in  $B$ :

$$FT_{B, C_1}^A = S_{C_1}^{C_2} \otimes FT_{B, C_2}^A \quad (4)$$

$FT_{B, X}^A$  is extended notation for  $A$ 's functional trust to  $B$  which considers the underlying situation  $X$ . The higher the similarity between  $C_1$  and  $C_2$  is, the closer the trust of  $A$  to  $B$  in situation  $C_1$  will be equal to that of between  $A$  and  $B$  in situation  $C_2$ . The lower this similarity is, the more uncertain  $A$  will be about whether to trust  $B$  or not in the second situation.

The same conversion formula can be used for Referral Trust.

$$RT_{B, C_1}^A = S_{C_1}^{C_2} \otimes RT_{B, C_2}^A \quad (5)$$

<sup>2</sup> This metric is inferred from a metric for the trust value computation [13] by Jøsang and Knapskog.

## 4 Evaluation

We chose MovieLens data<sup>3</sup> in view of the fact that we needed a context-enriched data to evaluate our work. The MovieLens data has been collected by the GroupLens Research Project at the University of Minnesota<sup>4</sup>. The data consists of 100,000 ratings from 943 users on 1682 movies with every user having at least 20 ratings and simple demographic information for the users is included. Figure 5 depicts the ontology which corresponds to the MovieLens data.

User attributes are age, sex and 19 occupation categories<sup>5</sup>, zipcode, and movie attributes are 19 film genres<sup>6</sup>. Much richer movie content can be obtained from the Internet Movie Database (IMDB)<sup>7</sup>. We consider user and movie concepts as contexts and user and movie attributes as local contexts to form the situation for each rating.

### 4.1 Data Setup

There are 5 datasets which are 80%/20% splits of the data into training and test data (training set of 80,000 ratings, and the test set of 20,000 ratings). Each of these datasets have disjoint test sets; this is for 5 fold cross validation (where we repeat our experiment with each training and test set and average the results). The test sets are used as references for the accuracy of the predictions.

In the MovieLens data, rating values 1 and 2 represent negative ratings, 4 and 5 represent positive ratings, and 3 indicates ambivalence (we consider them as -2,-1,0,+1,+2). In order to convert these rating values to the Subjective Logic opinions (the triple  $(b, d, u), b + d + u = 1$ ) we can use the following conversion method:

$$b = \frac{\sum_{i=2}^n (i-1) \cdot f(i)}{c + (n-1) \cdot \sum_{i=1}^n f(i)}, \quad d = \frac{\sum_{i=1}^{n-1} (n-i) \cdot f(i)}{c + (n-1) \cdot \sum_{i=1}^n f(i)}, \quad u = \frac{c}{c + (n-1) \cdot \sum_{i=1}^n f(i)} \quad (6)$$

where the number of ratings at level  $i$  is described by function  $f(i)$  and  $c$  is a constant.

### 4.2 Experimental Setup

The casebase is built up from the ratings in the training set. Each case is composed of four parts: user identifier, movie identifier, rating value, and situation including user and movie information. Ratings in the test set forms queries to CMF and each query is composed of three parts: user identifier, movie identifier, and the situation (the rating

<sup>3</sup> <http://www.grouplens.org/node/73>

<sup>4</sup> <http://www.cs.umn.edu/Research/GroupLens/data/>

<sup>5</sup> Occupation list: administrator, artist, doctor, educator, engineer, entertainment, executive, healthcare, homemaker, lawyer, librarian, marketing, none, other, programmer, retired, salesman, scientist, student, technician, writer.

<sup>6</sup> Film genres: unknown, action, adventure, animation, children, comedy, crime, documentary, drama, fantasy, film-noir, horror, musical, mystery, romance, sci-fi, thriller, war, western.

<sup>7</sup> <http://us.imdb.com>



value is removed). The rating value in the query is predicted by CMF using the casebase, and then consequently compared with the removed value in the test set.

Four types of evaluation criteria are used in this paper:

- Coverage: measure of the percentage of movies in the test dataset that can be predicted.
- FCP: fraction of correct predictions.
- MAE (Mean Absolute Error) : average of the prediction error (difference between probability expected values of predicted and real opinions) over all queries.
- RMSE (root mean squared error) : root mean of the average of the squared prediction error. RMSE tends to emphasize large errors.

The evaluation is described as a pseudo-code in algorithm 1. First, the casebase and the set of queries are built from training and test sets, respectively. Second, the *Remember* procedure is called for each query computes the similarity between each case in the casebase and the query. Cases with a similarity less than a threshold are ignored and the ten most similar cases among the remainings are retrieved. Next, by calling the *Reuse* procedure, a rating value is predicted for the query ( $R_q$ ) based on the rating values of the retrieved cases ( $R_i, i = 1..10$ ) and their similarity measures ( $S_i$ ) which are calculated by the *Similarity* procedure.

$$R_q = (S_1 \otimes R_1) \oplus (S_2 \otimes R_2) \oplus \dots \oplus (S_{10} \otimes R_{10}) \quad (7)$$

Then, a new case is built which contains user and movie information of the query and the predicted rating value is added to the casebase by calling the *Learn* procedure. The predicted ratings form the *predicted set*. Finally, the *test* and *predicted* sets are compared according to the four metrics (Coverage, FCP, MAE, and RSME) by calling the *Evaluate* procedure.

The *Similarity* procedure (see algorithm 2) calculates weighted average of similarity measures of local contexts (age, sex, occupation, and zipcode for users and genres for movies) to determine the similarity between situations. In our implementation these weights are 0.2, 0.15, 0.1, 0.05, 0.5 respectively and are determined based on the fact that how much the local context can affect the rating decision. The comparator for each local context are:

- Age: Closer values are more similar.
- Sex: The similarity value is 1 for identical sex values and 0 otherwise.
- Occupation: The similarity is calculated according to (2) for similarity measurement on the ontology.
- Zipcode: ZIP codes are numbered with the first digit representing a certain group of U.S. states, the second and third digits together representing a region in that group (or perhaps a large city) and the fourth and fifth digits representing a group of delivery addresses within that region. We assign similarity values of 1, 0.75, 0.5 to the same delivery address, region, and state group respectively.
- Movie genre: The similarity is calculated using (2) to measure similarity on the ontology.

Our baseline is the Pearson algorithm [17] which relies on Pearson correlation coefficient to produce a correlation metric between users. This correlation is then used to weigh the rating of each relevant user. The Pearson correlation between users  $A$  and  $B$  is defined as:

$$P_{A,B} = \frac{\sum_{i=1}^m (R_{A,i} - \bar{R}_A) \times (R_{B,i} - \bar{R}_B)}{\sigma_A \times \sigma_B} \quad (8)$$

---

**Algorithm 1.** CONTEXT MANAGEMENT FRAMEWORK( $test\_set, training\_set$ )

---

**main**

**global**  $casebase, similarity$

**comment:** Build “casebase” from the training set and “queries” from the test set

$similarity[1..size(casebase)] \leftarrow 0$

**comment:** “similarity” array stores similarity measures between the query and the cases

**for each**  $query \in queries$

**do**  $\left\{ \begin{array}{l} neighbors \leftarrow \text{REMEMBER}(query, casebase) \\ predicted\_rating \leftarrow \text{REUSE}(neighbors) \\ \text{LEARN}(query, predicted\_rating) \\ predicted\_set \leftarrow predicted\_set \cup predicted\_rating \end{array} \right.$

EVALUATE( $test\_set, predicted\_set$ )

**procedure** REMEMBER( $query$ )

**for each**  $case \in casebase$

**do**  $\left\{ \begin{array}{l} sim \leftarrow \text{SIMILARITY}(query, case) \\ \text{if } sim \geq THRESHOLD \\ \quad \text{then } similarity[case] \leftarrow sim \end{array} \right.$

**return** (ten most similar cases)

**procedure** REUSE( $neighbors$ )

$predicted\_opinion \leftarrow (0, 0, 1)$

**for each**  $ncase \in neighbors$

**do**  $\left\{ \begin{array}{l} similarity\_opinion \leftarrow (similarity[ncase], 0, 1 - similarity[ncase]) \\ new\_opinion \leftarrow similarity\_opinion \otimes ncase.rating \\ predicted\_opinion \leftarrow predicted\_opinion \oplus new\_opinion \end{array} \right.$

**return** ( $predicted\_opinion$ )

**procedure** LEARN( $query, predicted\_rating$ )

$new\_case \leftarrow query.user \cup query.movie \cup predicted\_rating$

$casebase \leftarrow casebase \cup new\_case$

**procedure** EVALUATE( $test\_set, predicted\_set$ )

$coverage \leftarrow$  fraction of predicted ratings

$fcp \leftarrow$  fraction of correct predictions

$mae \leftarrow$  mean absolute error of predictions

$rmse \leftarrow$  root mean squared error of predictions

**output** ( $coverage, fcp, mae, rmse$ )

---

---

**Algorithm 2.** SIMILARITY(*query, case*)
 

---

```

procedure SIMILARITY(query, case)
    userq ← query.user
    userc ← case.user
    agesim ←  $1 - \frac{age_q - age_c}{age_{max} - age_{min}}$ 
    if sexq == sexc
        then sexsim ← 1
    else sexsim ← 0
    occupationsim ← ONTOLOGYSIM(occupationq, occupationc)
    comment: “OntologySim” calculates contextual similarity according to (2)
    if zipcodeq(1) == zipcodec(1)
        then
            if zipcodeq(2,3) == zipcodec(2,3)
                then
                    if zipcodeq(4,5) == zipcodec(4,5)
                        then
                            zipcodesim ← 1
                            comment: the same delivery address
                        else
                            zipcodesim ← 0.75
                            comment: the same region
                    else
                        zipcodesim ← 0.5
                        comment: the same state group
                else
                    zipcodesim ← 0
            else
                moviesim ← ONTOLOGYSIM(movieq.genre, moviec.genre)
                totalsim ← 0.2 · agesim + 0.15 · sexsim + 0.1 · occupationsim
                + 0.05 · zipcodesim + 0.5 · moviesim
        return (totalsim)
    
```

---

where  $m$  is the number of movies that both users rated.  $R_{A,i}$  is the rating, user  $A$  gave to movie  $i$ .  $\bar{R}_A$  is the average rating user  $A$  gave to all movies, and  $\sigma_A$  is the standard deviation of those ratings. Once the Pearson correlation between a user and all other users is obtained, the predicted movie rating is calculated as:

$$R_{A,i} = \bar{R}_A + \frac{\sum_{U=1}^n (R_{U,i} - \bar{R}_U) \times P_{A,U}}{\sum_{u=1}^n |P_{A,U}|} \quad (9)$$

Use of the Pearson correlation coefficient is quite common in the field of collaborative filtering, and results obtained with this method will be used to gauge the performance of other algorithms. Moreover, the Pearson algorithm uses only the rating information while our method use situational information to do the prediction.

### 4.3 Discussion of the Obtained Results

In table 1, we present the final results of the evaluation. We start by commenting the row “Coverage”. The coverage becomes an important issue on a very sparse dataset

**Table 1.** Final evaluation results

Metric	Dataset1	Dataset2	Dataset3	Dataset4	Dataset5	Average	Pearson CC
Coverage (%)	43.82	43.88	44.94	45.42	45.06	44.62	99.83
FCP	0.3629	0.3497	0.3299	0.3345	0.3417	0.3437	0.1993
MAE	0.1605	0.1600	0.1656	0.1648	0.1626	0.1627	0.3049
RMSE	0.2742	0.2717	0.2757	0.2739	0.2724	0.2736	0.3804

that contains a large portion of cold-start users since many trust values become hardly predictable [18]. The results (*Coverage*  $\approx 0.45\%$ ) indicate that our model is able to predicate approximately one rating from each two ratings. For the Pearson algorithm the coverage is not perfect merely because not all movies in the test dataset have a rating in the training dataset. The second important result is the fraction of correct predictions (FCP) is 0.34 which shows that from each 10 predicted ratings between 3 and 4 ratings are predicted with exact values. Further, the prediction errors (MAE and RMSE) for the other ratings that are not predicted exactly (between 6 and 7 ratings from each 10 predicted ratings) are small in comparison with the Pearson method (*MAE*  $\approx 0.12$  & *RMSE*  $\approx 0.20$ ).

All-in-all, the results of the evaluation lead to the expectation that our approach provides an improvement over the Pearson algorithm and this implies that situational information is useful in making predictions.

## 5 Related Research

CMF is a *knowledge-intensive CBR* which is designed to extend situational inference capabilities of *trust* management models. More precisely, the aim is to reuse the available trust information (direct experiences and recommendations) in similar situations for the current problem and we use semantic (ontology-based) similarity measures. Although CBR techniques are extensively used for recommender systems [1, 24] and there are some works which use CBR to build more trust through providing explanations [16, 21, 22], to the best of our knowledge this proposal is quite new. In this section, we briefly explain the related researches which are based on context-aware trust management and thus more closely resemble our goal.

According to the literature, the extension of a trust model with context representation can reduce complexity in the management of trust relationships [20], improve the recommendation process [20], help to infer trust information in context hierarchies [9], improve performance [25], help to learn policies/norms at runtime [25, 29], and provide protection against changes of identity and first time offenders [25]. Context related information has been represented as Context-aware domains [20], Intensional Programming [31], Multi-dimensional goals [8], Clustering [25], and Ontologies [29].

[26] provides a survey of different approaches to model context for ubiquitous computing. In particular, numerous approaches are reviewed, classified relative to their core elements and evaluated with respect to their appropriateness for ubiquitous computing.

The authors conclude that the most promising assets for context modeling of ubiquitous computing environments can be found in the ontology category in comparison with other approaches like key-value models, mark-up scheme models, graphical models, object-oriented models, and logic based models. This selection is based on the six requirements dominant in pervasive environments: distributed composition, partial validation, richness and quality of information, incompleteness and ambiguity, level of formality, and applicability to existing environments.

We present a state-of-the-art survey of context representation for trust management in [28]. In the rest of this section ontology-based approaches to this problem are examined in more details.

Golbeck et al. [6] propose an ontology for trust. In [5] the authors consider a model using context-specific reputation by assigning numeric ratings to different types of connections based on context of the analysis. In [29] rules to describe how certain context-sensitive information (trust factors) reduces or enhances the trust value have been specified for this trust ontology.

In [29] contextual information (i.e., context attributes) is used to adjust the output of a trust determination process. Each attribute can adjust the trust value positively or negatively according to a specified weight. As an illustration, if  $t$  is the trust value and  $\omega$  is the weight of the context property then the adjusting function can be  $t^\omega$  for decrease or  $\sqrt[\omega]{t}$  for increase. A context ontology connects the context attributes with each other in an appropriate manner, enabling the utilization of context attributes which do not exactly match the query, but are “close enough” to it.

In [2], cases where a trustor does not have enough information to produce a trust value for a given task, but she knows instead the previous partner behavior performing similar tasks, are considered. This model estimates trust using the information about similar tasks. The similarity between two tasks is obtained from the comparison of the task attributes.

## 6 Conclusion and Future Directions

To sum up, we propose a framework based on the case-based reasoning paradigm and the representation of deep knowledge to make existing trust management models situation-aware. This framework has been validated for the Subjective Logic trust management model as an example and evaluated using a real large-scale dataset. It can also be considered as an inference mechanism which deals with the sparsity and cold-start problems of a web of trust.

The original Subjective Logic can be applied to determine transitivity only if the subject of the trust relations along the entire path is the same. However, trust relations with the same subject are not always available. Our proposal opens up the possibility to draw transitivity also when the subject (situation) of the available trust relations are not the same but are similar. First, the trust relations with similar situations with the current problem are retrieved from the casebase using the ontology and the similarity measurement algorithm (remembering past similar trust experiences). Next, they are converted (using (4) and (5)) to equivalent trust relations in the current problem by solution transformation module (reusing the trust information from the past similar trust

experiences). Then, the transitive trust path is formed and final trust is calculated according to the Subjective Logic (II). Solution of the current problem is stored as a new case in the casebase (the learning process of CBR).

In the future, we aim to add a Risk Management Module to this framework. Risk evaluation becomes important in inferring trust values among situations especially when the trustworthiness of some principal is completely unknown and no recommendation information is available. The intuitive idea behind such a risk assessment can be to look up the in the casebase to see if there are any similar previous interactions, i.e., if we have previously encountered an entity with similar trust attributes and similar risk attributes in the same situation. The ontology part should be able to describe the level of situational risk, whereby the higher the risk of negative outcome, the higher the level of precision that must be captured.

## References

1. Aguzzoli, S., Avesani, P., Massa, P.: Collaborative Case-Based Recommender Systems. In: Craw, S., Preece, A.D. (eds.) ECCBR 2002. LNCS, vol. 2416, pp. 460–474. Springer, Heidelberg (2002)
2. Caballero, A., Botía, J.A., Gómez-Skarmeta, A.F.: On the Behaviour of the TRSIM Model for Trust and Reputation. In: Petta, P., Müller, J.P., Klusch, M., Georgeff, M. (eds.) MATES 2007. LNCS, vol. 4687, pp. 182–193. Springer, Heidelberg (2007)
3. Christianson, B., Harbison, W.S.: Why Isn't Trust Transitive? In: Lomas, M. (ed.) Security Protocols 1996. LNCS, vol. 1189, pp. 171–176. Springer, Heidelberg (1997)
4. Ding, L., Kolari, P., Ganjugunte, S., Finin, T., Joshi, A.: Modeling and Evaluating Trust Network Inference. Technical report, Maryland Univ. Baltimore Dept. of Computer Science and Electrical Engineering (2005)
5. Golbeck, J., Hendler, J.: Inferring Reputation on the Semantic Web. In: Proceedings of the 13th International World Wide Web Conference (2004)
6. Golbeck, J., Parsia, B., Hendler, J.: Trust Networks on the Semantic Web. In: Klusch, M., Omicini, A., Ossowski, S., Laamanen, H. (eds.) CIA 2003. LNCS, vol. 2782, pp. 238–249. Springer, Heidelberg (2003)
7. Guha, R., Kumar, R., Raghavan, P., Tomkins, A.: Propagation of trust and distrust. In: Proceedings of the 13th international conference on World Wide Web, pp. 403–412. ACM Press, New York (2004)
8. Gujral, N., DeAngelis, D., Fullam, K.K., Barber, K.S.: Modeling Multi-Dimensional Trust. In: The Proceedings of the Workshop on Trust in Agent Societies, pp. 8–12 (2006)
9. Holtmanns, S., Yan, Z.: Context-Aware Adaptive Trust
10. Jøsang, A.: A Logic for Uncertain Probabilities. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 9(3), 279–311 (2001)
11. Jøsang, A.: The consensus operator for combining beliefs. *Artificial Intelligence* 141(1-2), 157–170 (2002)
12. Jøsang, A., Hayward, R., Pope, S.: Trust network analysis with subjective logic. In: Proceedings of the 29th Australasian Computer Science Conference, vol. 48, pp. 85–94. Australian Computer Society (2006)
13. Jøsang, A., Knapskog, S.J.: A metric for trusted systems. In: Proceedings of the 21st National Security Conference, NSA (1998)
14. Josang, A., Marsh, S., Pope, S.: Exploring Different Types of Trust Propagation. In: Stølen, K., Winsborough, W.H., Martinelli, F., Massacci, F. (eds.) iTrust 2006. LNCS, vol. 3986, pp. 179–192. Springer, Heidelberg (2006)

15. Jung, C., Han, I., Suh, B.: Risk Analysis for Electronic Commerce Using Case-Based Reasoning. *Int. J. Intell. Sys. Acc. Fin. Mgmt.* 8, 61–73 (1999)
16. Leake, D.B.: CBR in Context: The Present and Future. *Case-Based Reasoning: Experiences, Lessons, and Future Directions*, 3–30 (1996)
17. Massa, P., Avesani, P.: Trust-Aware Collaborative Filtering for Recommender Systems. In: Meersman, R., Tari, Z. (eds.) *OTM 2004. LNCS*, vol. 3290, pp. 492–508. Springer, Heidelberg (2004)
18. Massa, P., Avesani, P.: Trust-aware recommender systems. In: *Proceedings of the 2007 ACM conference on Recommender systems*, pp. 17–24. ACM Press, New York (2007)
19. Morris, B.W.: SCAN: a case-based reasoning model for generating information system control recommendations. *International Journal of Intelligent Systems in Accounting, Finance and Management* 3(1), 47–63 (1994)
20. Neisse, R., Wegdam, M., van Sinderen, M., Lenzini, G.: Trust Management Model and Architecture for Context-Aware Service Platforms. In: Meersman, R., Tari, Z. (eds.) *OTM 2007, Part II. LNCS*, vol. 4804, pp. 1803–1820. Springer, Heidelberg (2007)
21. Pu, P., Chen, L.: Trust building with explanation interfaces. In: *Proceedings of the 11th international conference on Intelligent User Interfaces*, pp. 93–100. ACM, New York (2006)
22. Pu, P., Chen, L.: Trust-inspiring explanation interfaces for recommender systems. *Knowledge-Based Systems* 20(6), 542–556 (2007)
23. Quercia, D., Hailes, S., Capra, L.: Lightweight Distributed Trust Propagation. In: *Seventh IEEE International Conference on Data Mining, ICDM 2007*, pp. 282–291 (2007)
24. Recio-García, J.A., Bridge, D.G., Díaz-Agudo, B., González-Calero, P.A.: CBR for CBR: A case-based template recommender system for building case-based systems. In: Althoff, K.-D., Bergmann, R., Minor, M., Hanft, A. (eds.) *ECCBR 2008. LNCS*, vol. 5239, pp. 459–473. Springer, Heidelberg (2008)
25. Rehak, M., Pechoucek, M.: Trust modeling with context representation and generalized identities. In: Klusch, M., Hindriks, K.V., Papazoglou, M.P., Sterling, L. (eds.) *CIA 2007. LNCS*, vol. 4676, pp. 298–312. Springer, Heidelberg (2007)
26. Strang, T., Linnhoff-Popien, C.: A context modeling survey. In: *Workshop on Advanced Context Modelling, Reasoning and Management as part of UbiComp (2004)*
27. Tavakolifard, M., Knapskog, S., Herrmann, P.: Cross-Situation Trust Reasoning. In: *Proceedings of The Workshop on Web Personalization, Reputation and Recommender Systems (WPRRS 2008)*. IEEE Computer Society Press, Los Alamitos (2008)
28. Tavakolifard, M., Knapskog, S., Herrmann, P.: Trust Transferability Among Similar Contexts. In: *Proceedings of The 4th ACM International Workshop on QoS and Security for Wireless and Mobile Networks (Q2SWinet 2008)*. ACM, New York (2008)
29. Toivonen, S., Lenzini, G., Uusitalo, I.: Context-aware trust evaluation functions for dynamic reconfigurable systems. In: *Proceedings of the Models of Trust for the Web Workshop (MTW 2006)*, held in conjunction with the 15th International World Wide Web Conference (WWW 2006), May 2006, vol. 22 (2006)
30. Tversky, A., et al.: Features of similarity. *Psychological Review* 84(4), 327–352 (1977)
31. Wan, K., Alagar, V.: An Intensional Functional Model of Trust. In: *Proceedings of IFIPTM 2008 - Joint iTrust and PST Conferences on Privacy, Trust Management and Security*, pp. 69–85. Springer, Boston (2008)

# TIUPAM: A Framework for Trustworthiness-Centric Information Sharing

Shouhuai Xu<sup>1</sup>, Ravi Sandhu<sup>2</sup>, and Elisa Bertino<sup>3</sup>

<sup>1</sup> Department of Computer Science, Univ. of Texas at San Antonio  
shxu@cs.utsa.edu

<sup>2</sup> Institute for Cyber Security, Univ. of Texas at San Antonio  
ravi.sandhu@utsa.edu

<sup>3</sup> Department of Computer Science, Purdue University  
bertino@cs.purdue.edu

**Abstract.** Information is essential to decision making. Nowadays, decision makers are often overwhelmed with large volumes of information, some of which may be inaccurate, incorrect, inappropriate, misleading, or maliciously introduced. With the advocated shift of information sharing paradigm from “need to know” to “need to share” this problem will be further compounded. This poses the challenge of achieving assured information sharing so that decision makers can always get and utilize the up-to-date information for making the right decisions, despite the existence of malicious attacks and without breaching privacy of honest participants. As a first step towards answering this challenge this paper proposes a systematic framework we call TIUPAM, which stands for “Trustworthiness-centric Identity, Usage, Provenance, and Attack Management.” The framework is centered at the need of trustworthiness and risk management for decision makers, and supported by four key components: identity management, usage management, provenance management and attack management. We explore the characterization of both the core functions and the supporting components in the TIUPAM framework, which may guide the design and realization of concrete schemes in the future.

## 1 Introduction

Information sharing is an important process in human society because it helps make better decisions. However, information should not be arbitrarily disseminated for various reasons including sensitivity and privacy, and truthfulness of information should not be taken for granted. The latter is especially important in adversarial environments, such as business, economics, and military. Traditionally, the research communities and the industrial vendors have focused on enforcing “need to know” via various mechanisms. Recently, a new paradigm known as “need to share” has emerged, primarily to more effectively deal with threats such as terrorist attacks and demonstrated failure of “need to know” in this regard. This brings new challenges because (1) decision makers are potentially even more overwhelmed with information, which should by no means be treated as trustworthy, and (2) the “access control”-centric solution paradigm is not sufficient anymore because the notions of *authorization* and *authentication* are less



explicit. In this paper we propose a solution framework to help the decision makers deal with this new challenge.

**Our contributions.** We propose a systematic framework called TIUPAM, which stands for “Trustworthiness-centric Identity, Usage, Provenance, and Attack Management”. The framework is centered at serving decisionmakers’ needs for effectively managing the trustworthiness of information as well as the risk that may be caused by utilizing or not utilizing available information. This core of trustworthiness and risk management is supported by four components.

- *Identity management.* Identity management serves trustworthiness and risk management, provenance management, as well as usage management while receiving services from provenance management and usage management. In particular, it allows the participants to evaluate the trustworthiness of the digital identities and digital credentials for people, organizations, and devices.
- *Usage management.* Usage management serves trustworthiness and risk management while receiving services from identity management. It mainly deals with authorized activities. It extends current generation of usage control by considering, for example, the trustworthiness of both requests and information.
- *Provenance management.* Provenance management serves trustworthiness and risk management by essentially enabling the evaluation of trustworthiness of data, software, and requests, while receiving service from identity management.
- *Attack management.* Attack management serves all of the aforementioned components by dealing with attacks and unauthorized activities. In particular, the evaluation of trustworthiness of information, identity, request, usage, and provenance must be with respect to some specific attack model.

The focus of this paper is on exploration of the characteristics of the core functions and components, rather than specifying any concrete realizations. This characterization would help the design of concrete schemes for realizing the framework in the future.

**Paper organization.** Section 2 presents the TIUPAM framework as well as its core functions. Section 3 discusses the identity management component, Section 4 presents the usage management component, Section 5 discusses the provenance management component, and Section 6 presents the attack management component. Section 7 summarizes the paper.

## 2 The TIUPAM Framework

Within this framework and throughout the present paper, we use the term “information” in a broad sense, meaning that it accommodates information items, data items, message items, and knowledge items.

### 2.1 Framework Components and Their Logical Relationships

As illustrated in Figure 1, the TIUPAM framework is centered at trustworthiness and risk management, which serves the need of decision makers. The supporting components are identity management, usage management, provenance management, and attack management, whose logical relationships are depicted in Figure 2.

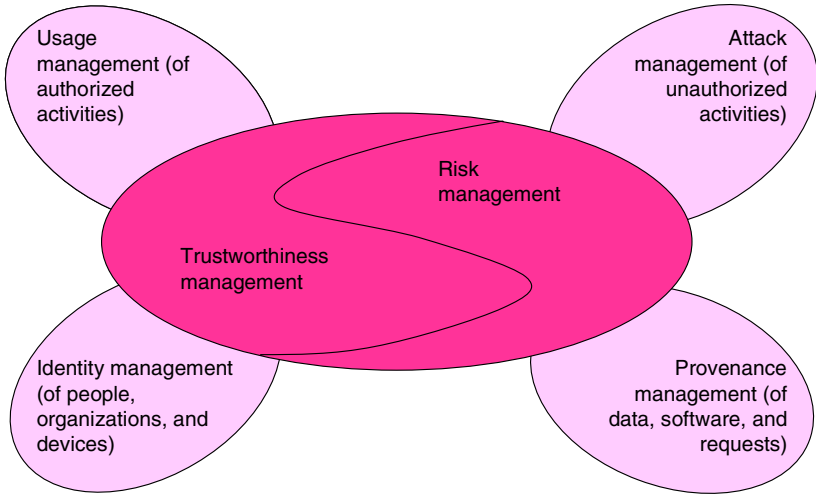


Fig. 1. Key components of the TIUPAM framework

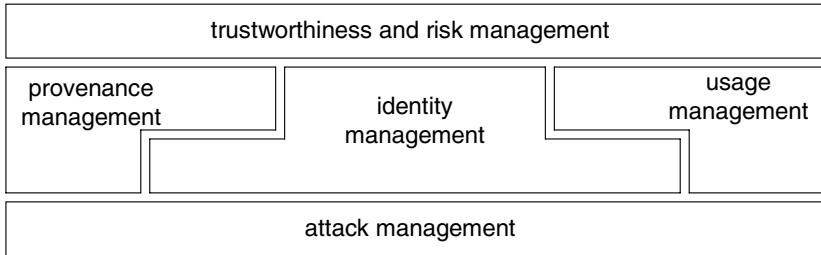


Fig. 2. Logical relationship between the components

In what follows we elaborate on the functionalities of the core as well as the supporting components.

- Trustworthiness and risk management:* For decision makers, the most important issue is the trustworthiness of the information at hand, which reflects the decision maker’s current “snapshot” of the world and may be (in)accurate, (in)correct, misleading, or even maliciously introduced. The term “snapshot” is emphasized because, in the context of the present paper, trustworthiness is meant to capture the *dynamical* evaluation of the degree of information being trustable or trustworthy. The term “dynamical” indicates that one’s evaluation of trustworthiness of some information may change with respect to time, as more information is gathered. (In contrast, trust can be invariant regardless of the information currently available; for example, we may still trust an individual even if there is information or rumors against that person.) Corresponding to the non-perfect trustworthy information, any decision based on the “snapshot” bears some risk because its execution

may lead to negative consequences. We stress that trustworthiness is a measure against the snapshot of one's up-to-date observation about the information in question; whereas, risk is a measure against the potential consequences caused by the execution of decisions based on not-necessarily-trustworthy information, based on the state-of-the-art understanding of the world. Therefore, trustworthiness and risk are not necessarily complementary to each other.

- *Identity management*: Identity, including digital credentials, provides a base for trustworthiness, risk, provenance, and usage management. Specifically, in order to support trustworthiness and risk management, we need to measure the trustworthiness of identities of people, organizations, and devices. This is because the aforementioned snapshots are derived, in one way or another, from the statements asserted by the relevant people, organizations, and devices. For example, a software program digitally signed by a software vendor may certify that the output corresponding to a given input to the program is indeed the desired result (e.g., some knowledge extracted from data with respect to the algorithm the program executes); a message digitally signed by an organization would make one tend to believe its trustworthiness; a successful attestation of a remote device may lead us to accept that the remote peering computer is not compromised.
- *Usage management*: Usage management seeks to manage authorized activities by extending traditional access control. It was inspired by the following observations on the limitation of traditional access control: (1) a subject is always trustworthy as long as it passes certain pre-determined authentication, and (2) an object is always trustworthy as long as it is in the filesystem or database. The former preassumption is faulty if the authentication credential of the subject has been compromised, and the later preassumption is faulty if the object itself was malicious or incorrectly provided. Therefore, it is important for usage management to take into account, among other things, the trustworthiness of both data and requests, which in turn requires to take into account the trustworthiness of both provenance and identity.
- *Provenance management*: Provenance management directly serves the higher-layer trustworthiness and risk management by managing the provenance of information, software, and requests etc, while being served by identity management and usage management. Provenance of data allows us to measure the trustworthiness of information; provenance of software helps to evaluate the trustworthiness of software programs; provenance of requests enhances the assurance of the requests' source in that they are invoked by the individual or process in question, rather than by malware.
- *Attack management*: Attack management deals with unauthorized activities, especially malicious attacks that may intentionally introduce wrong or misleading information into the system. In particular, it helps manage the trustworthiness of infrastructure-level services provided to the other components as well as their services in the framework (e.g., authentication services). This is an important problem and more subtle than first glance because traditionally people tend to accept that infrastructures (e.g., public key infrastructures or PKI) as trustworthy simply because of their absolute trust in them.

## 2.2 Core Functions of Trustworthiness and Risk Management

Figure 2 highlighted the logical relationships between the components. In what follows we discuss the core functions that should be realized by the TIUPAM framework to the applications. The relationships between the functions are highlighted in Figure 3 and elaborated below.

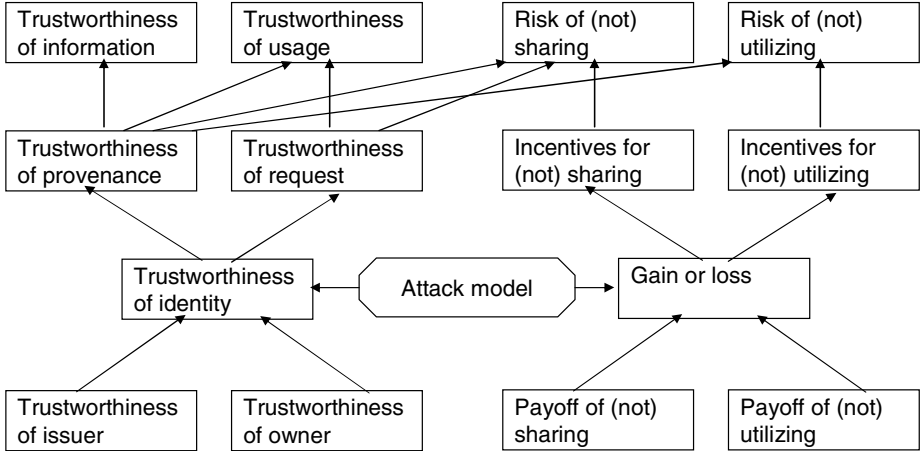


Fig. 3. Structures of the trustworthiness and risk functions (attack models are not elaborated for a better visual effect)

Decisionmakers would often need to resolve the following questions: How trustworthy is a given information? How trustworthy is the adherence of an information consumer to the usage policy? What is the risk incurred by sharing or not sharing a certain information? What is the risk because of utilizing or not utilizing a given information? Corresponding to these questions, we may define the following function families.

- **Trustworthiness of information:** It should be a function of the trustworthiness of the provenance of the information in question. Therefore, there are families of functions  $\{f_1\}$ ,  $\{f_{11}\}$ , and  $\{f_{111}\}$  such that

$$\begin{aligned}
 \text{trustworthiness\_of\_data} &= f_1(\text{trustworthiness\_of\_provenance}), \text{ where} \\
 \text{trustworthiness\_of\_provenance} &= f_{11}(\text{trustworthiness\_of\_identity}), \text{ and} \\
 \text{trustworthiness\_of\_identity} &= f_{111}(\text{trustworthiness\_of\_issuer}, \\
 &\text{trustworthiness\_of\_owner}, \text{attack\_model}).
 \end{aligned}$$

Note that the *attack\_model* is always an input argument to some “low level” functions, meaning that it is implicit in the “high level” functions such as *trustworthiness\_of\_data*. The motivation is that in order to evaluate the functions in a consistent fashion, the same *attack\_model* should be used in the bottom-up evaluation of the functions.

- **Trustworthiness of usage:** It should be a function of both trustworthiness of provenance and trustworthiness of request. Therefore, there are families of functions  $\{f_2\}$  and  $\{f_{21}\}$  such that

$trustworthiness\_of\_usage = f_2(trustworthiness\_of\_provenance,$   
 $trustworthiness\_of\_request)$ , where  
 $trustworthiness\_of\_request = f_{21}(trustworthiness\_of\_identity)$ , and  
 $trustworthiness\_of\_identity = f_{111}(trustworthiness\_of\_issuer,$   
 $trustworthiness\_of\_owner, attack\_model)$ .

- **Risk of sharing information:** It should be a function of the trustworthiness of the provenance of the information in question, the trustworthiness of request (which may be explicit in pull-based information sharing and implicit in push-based information sharing), and incentives for sharing. Therefore, there are a family of functions  $\{f_3\}$ ,  $\{f_{31}\}$ ,  $\{f_{311}\}$  such that

$risk\_of\_sharing\_information = f_3(trustworthiness\_of\_provenance,$   
 $trustworthiness\_of\_request, incentive\_for\_sharing\_information)$ , where  
 $incentive\_for\_sharing\_information = f_{31}(gain\_because\_of\_sharing)$ , and  
 $gain\_because\_of\_sharing = f_{311}(payoff\_of\_sharing, attack\_model)$ .

Note that  $gain\_because\_of\_sharing$  must take  $attack\_model$  into consideration because in different attack models the outcome can be completely opposite (e.g., the information receiver is truly the claimed authorized user vs. the information receiver is actually the attacker who can perfectly impersonate the user because the attacker has compromised the user's identity). Similarly, we can define the function families for specifying the risk of not sharing.

- **Risk of utilizing received information:** It should be a function of the trustworthiness of information provenance and the incentives for utilizing the information in question. Therefore, there are a family of functions  $\{f_4\}$ ,  $\{f_{41}\}$ ,  $\{f_{411}\}$  such that

$risk\_of\_utilizing\_information = f_4(trustworthiness\_of\_provenance,$   
 $incentive\_for\_utilizing\_information)$ , where  
 $incentive\_for\_utilizing\_information = f_{41}(gain\_because\_of\_utilizing)$ ,  
 $gain\_because\_of\_utilizing = f_{411}(payoff\_of\_utilizing, attack\_model)$ .

Similarly, we can define the function families for specifying the risk of not utilizing a given information (e.g., because of its low or uncertain degree of trustworthiness).

It should be noted that risk comes from two aspects: (1) the consequences that may be caused by utilizing incorrect or malicious information (e.g., because it is accompanied with a high degree of trustworthiness); (2) the consequences that may be caused by not utilizing the not-known-to-be, but indeed trustworthy, information (e.g., because it is accompanied with a low or uncertain degree of trustworthiness). The risk will be evaluated whenever a relevant decision is being made; for example, whether to allow the use or exchange of information.

We reiterate that the components aim to evaluate and maintain the trustworthiness of information in a dynamic fashion because the trustworthiness should always be updated. For example, we may treat a data item as fully trustworthy today even though it was only partially trustworthy yesterday because of new insights obtained since then.

We emphasize that it is not our aim in this paper to define the function families mentioned above, which should be specific to the applications. Rather, we want to clearly state the framework by detailing the relationship between the components.

### 3 Identity Management

In this section we discuss the key properties of desired identity management systems.

**Extensibility.** Any good identity management should be easily extended to accommodate or integrate emerging new identity systems. This is important because as the computing environments evolve, so do the individual identity management systems. Moreover, the diversity of applications often implies diversity in digital identity or credential systems. This is important because while we are used to digital identities such as public keys or attribute certificates, facilitated by a public key infrastructure (PKI), other types of identities may emerge. For example, in the case of mobile computing, two users with no common trusted third party could establish a mutual trust on their own. Moreover, this individual trust may further bootstrap future trust establishment between their friends because social networks are becoming an indispensable part of future computing paradigms. In turn, this means that future identity management systems should be easily extensible.

**Automated trustworthiness.** A key support of identity management systems to trustworthiness and risk management, usage management, and provenance management is the trustworthiness a verifier can put on a digital identity in question. This requires the identity management component to provide automated trustworthiness service by ensuring the following.

- *Compromise containment.* This states that the consequences due to the compromise of some computers or identities are contained and, ideally, minimized. There are several typical scenarios.
  - *Compromise of servers that authenticate users through their identities or credentials is contained.* This is relevant when the authentication is based on symmetric cryptography, including symmetric key cryptosystems and passwords. In this case, compromising a server could cause the compromise of the users' authenticators directly (e.g., when symmetric keys are used) or indirectly (e.g., after launching off-line dictionary attack when passwords are used). This is also relevant when the authentication is based on asymmetric cryptography, such as when servers store the public keys of users. For example, the attacker could tamper with the access history of the users, erase the access events incurred by the attacker, insert bogus user entries, or modify the public keys of the users. In all of these cases, the damage should be contained and, ideally, minimized.
  - *Compromise of some users' identities or credentials is contained to those users and, ideally, to those compromised (e.g., stolen) identities and credentials.* It is not unusual that every user has multiple digital identities and credentials, which may or may not be independent of each other at all (e.g., one user reuses a password for multiple accounts). There is a possibility that compromising a user's

computer could cause the compromise of all the user's identities or credentials, which corresponds to the worst-case scenario. Therefore, it is important to ensure containment in the following sense: Compromising of digital identity or credential for accessing one server does not cause the compromise of digital identity or credential for accessing another server.

- *Accountability.* Digital identities or credentials live and operate in a hostile environment wherein many computers, including well-protected servers, can be compromised. This puts in question accountability, enforcement of which deters many attacks. For example, if an access is launched through the use of stolen identity or credential, who should be held accountable for the consequences? It is arguable that the user, whose identity or credential was stolen, is a victim as well. Things could become much more complicated when, for example, a malicious user intentionally abuses this fact to hide its own unlawful activities. This calls for good forensics mechanisms to deter, if not absolutely hold the malicious users or attackers accountable for attacks and abuses.

## 4 Usage Management

The concept of usage control has recently emerged as a paradigm for next generation access control transcending the traditional access matrix model [2]. To accommodate modern applications, concepts such as trust management, digital rights management, obligations and attribute-based access control were proposed in the past decade. Usage control provides a unified framework for modeling these and other access-control extensions. Usage control maintains the classic access control abstraction of a right as a privilege that a subject must hold to access an object in different modes. Unlike the traditional access matrix, in usage control the existence of a right is determined when an access is attempted by a subject and may continue to be determined as the right is used. This usage decision is made based on subject attributes, object attributes, authorizations, obligations, and conditions. Specifically, authorizations are predicates that determine whether the subject (requester) holds the requested rights on the object, obligations are predicates that verify the subject has performed required actions prior or during the usage, and conditions are predicates on environmental or system state. Usage control explicitly recognizes a pre, ongoing and post phase for each usage of a resource. Another feature of usage control not present in conventional access control models is mutable attributes—attributes of subjects and attributes that are modified before, during, or after a usage session. Collectively these features allow for consumable rights and instant and preemptive revocation.

Considering the requirements of trustworthiness-centric information sharing discussed above, we identify two limitations of current usage control models, viz., future (or post) obligations and system obligations. For example, a physician accessing a patient's electronic health record in an emergency may have a pre-obligation to acknowledge that this is an emergency situation so that access is opened up. After the usage is completed, she may incur a post-obligation to file a statement confirming that the emergency access was justified. Even though the post-obligation occurs after access, it validates the circumstance of the completed access. Completion of the post-obligation

may be deferred into the future, allowing it to be done when the physician has some downtime. This then truly becomes a future obligation. The concept of a system obligation comes into play when an object is moved from one security domain to another. For example a document D from domain A could be made available in domain B but with policy requirements specified by domain A, such as make D accessible to no more than five users in domain B, store D in encrypted form and delete D after one month. These policy requirements are stated by domain A but enforced by domain B. We say domain B has an obligation to enforce the policy specified by domain A.

The dynamic characteristics of usage control are well suited to the problem of trustworthiness-centric information sharing. Usage control decisions are made at access time and continue to be revisited during access. As such the basic elements for dynamic decisions with respect to access are fundamental to usage control. However, trustworthiness and risk should be more explicitly incorporated into future versions and manifestations of usage control. While the current models for usage control provide the necessary foundational framework much research needs to be done to incorporate attributes and rules that effectively capture trustworthiness and risk as attributes.

## 5 Provenance Management

### 5.1 Functional Requirements

Without loss of generality, we assume that information may move within distributed/decentralized systems in the format of messages. Moreover, new messages may be produced by algorithms that may take other messages as inputs. That is, we are primarily dealing with information provenance management in distributed or decentralized systems, which might often be large-scale.

From a functional perspective, we believe that a secure provenance management system should cover the entire lifecycle of information as well as their associated provenance. In this context, we classify information lifecycle into the following procedures of generation and processing. We note that this lifecycle is somewhat tailored to secure provenance management systems, and thus may not be appropriate for other systems.

- *Generation*: An information item originally enters into a provenance management system through some participant; such participant is the party responsible for the initial generation and insertion of the information item into the system.
- *Processing*: Each participant, source or intermediate node, can produce new information items based on the items it received from other participants. Various (e.g., datamining or knowledge extraction) algorithms and functions are possible for producing information items. For example, such a function can simply consist of endorsing an information item another participant is disseminating.

### 5.2 Security Requirements

A secure provenance management system should provide information trustworthiness management service to higher layer applications. In general, information trustworthiness depends on the trustworthiness of the source, the trustworthiness of the intermediate nodes as well as their processing algorithms. However, things quickly become



complex when some participants (i.e., sources and intermediate nodes) may be malicious. In what follows we discuss some representative issues that relevant to how information trustworthiness should be managed.

- For a source, it is necessary to know about the trustworthiness of an information item that has to be entered into the system. It is also necessary that, when a source realizes that it has entered into the system inaccurate or even misleading information (e.g., deceptive information deliberately provided by adversary), the source be able to inform all the relevant participants about this fact (and possibly also to provide updated information).
- For an intermediate node, it is necessary to know about the trustworthiness of both the source and the prior intermediate nodes so that, for example, a decision may be made whether to re-disseminate the processed information. It is also important to allow a node to notify upstream nodes, e.g., that some information items they provided are inaccurate or even misleading (we may call this “backward information correction”), and to notify downstream nodes, e.g., that some information items they received are inaccurate or even misleading (we may call this “forward information correction”).
- For an information consumer, it is necessary to be able to evaluate the trustworthiness of an incoming information item. Moreover, the consumer must be cautious in making decisions that rely on such items because the decisions may not be reversible and, once enforced, may cause severe consequences.
- For an administrator, it is important to know who has a large influence or impact on the evolution of information in the networks? Enhancing security of such participants would significantly improve security from a whole-system perspective.

## 6 Attack Management

In order to enable trustworthiness and risk management, identity management, usage management, and provenance management, attack management seeks to systematically model the attacks against each of the relevant processes and procedures. Corresponding to Figure 3, we articulate the following attack models attempting to manipulate most of the functions. Note that the attacks accommodate those targeting application layer and those targeting infrastructure layer as well.

**Attacks attempting to manipulate the trustworthiness of information.** Trustworthiness of information can be manipulated by compromising the provenance of the information. There are several “attack points” at which the attacker can tamper with the provenance information. The attack points correspond to the generation of the information (e.g., a malicious user enters false information into the system), processing of the information (e.g., a malicious user claims that the information is the output of some legitimate application program), the dissemination of the information (e.g., a malicious user claims that the originator of the false information is a trustworthy source). One way to successfully launch the above attacks is to manipulate the trustworthiness of identities, which can be done by compromising and abusing the compromised

identity to impersonate the identity owner, or by compromising a victim identity issuer or becoming a malicious identity issuer.

**Attacks attempting to manipulate the trustworthiness of usage.** Trustworthiness of usage can be attacked by undermining the trustworthiness of information provenance so that, e.g., malicious information thereby spreads to a large population of users, or by manipulating the trustworthiness of request. The latter can be done by compromising the trustworthiness of identity (e.g., compromising the credential in question). It is also possible to manipulate trustworthiness of usage by attacking the management of *who could read/write/modify* as well as *who have read/written/modified* which information.

**Attacks attempting to manipulate the risk management.** Risk management can be undermined by manipulating the trustworthiness of the information in question (e.g., highly trustworthy information is deemed as low trustworthy, low trustworthy information is deemed as high trustworthy), or by manipulating the trustworthiness of the request (e.g., unauthorized users may become highly trustworthy in requesting the information).

**Attacks attempting to manipulate the privacy of honest participants.** Privacy is important in many applications and is relevant in all the components. Privacy protection may be at odds with trustworthiness because, for example, a malicious user may intentionally introduce misleading information into the system by abusing the anonymity protection shield so as to not be held accountable. Privacy protection can be dealt with by appropriate risk management in deciding whether or not to share some information, or whether or not to utilize some received information. Privacy protection is crucial to usage management because a malicious user may leak certain information without being held accountable.

## 7 Conclusion and Future Work

We have explored a systematic framework for trustworthiness-centric information sharing we called TIUPAM. Our framework consists of a core component — the trustworthiness and risk management, and four supporting components — identity management, usage management, provenance management, and attack management.

We highlighted the properties a desired solution should possess, and it is beyond the scope of the present paper for designing concrete solutions. As such, this paper introduces a range of challenging research problems for future work. For example, the identity and attack management explored in the paper is even more demanding than the state of the art in managing the trustworthiness of certain cryptographic credentials [3]; the provenance management explored in the paper is even more challenging than the provenance security discussed in [1].

**Acknowledgement.** This work is supported in part by AFOSR MURI award FA9550-08-1-0265.

## References

1. Braun, U., Shinnar, A., Seltzer, M.: Securing provenance. In: HotSec 2008 (2008)
2. Park, J., Sandhu, R.: The UCON ABC usage control model. *ACM Transactions on Information and System Security (TISSEC)* 7(1), 128–174 (2004)
3. Xu, S., Yung, M.: Expecting the unexpected: Towards robust credential infrastructure. In: FC (2009)

# TrustBuilder2: A Reconfigurable Framework for Trust Negotiation

Adam J. Lee<sup>1</sup>, Marianne Winslett<sup>2</sup>, and Kenneth J. Perano<sup>3</sup>

<sup>1</sup> Department of Computer Science, University of Pittsburgh  
adamlee@cs.pitt.edu

<sup>2</sup> Department of Computer Science, University of Illinois at Urbana-Champaign  
winslett@cs.uiuc.edu

<sup>3</sup> Sandia National Laboratories  
perano@ca.sandia.gov

**Abstract.** To date, research in trust negotiation has focused mainly on the theoretical aspects of the trust negotiation process, and the development of proof of concept implementations. These theoretical works and proofs of concept have been quite successful from a research perspective, and thus researchers must now begin to address the systems constraints that act as barriers to the deployment of these systems. To this end, we present TrustBuilder2, a fully-configurable and extensible framework for prototyping and evaluating trust negotiation systems. TrustBuilder2 leverages a plug-in based architecture, extensible data type hierarchy, and flexible communication protocol to provide a framework within which numerous trust negotiation protocols and system configurations can be quantitatively analyzed. In this paper, we discuss the design and implementation of TrustBuilder2, study its performance, examine the costs associated with flexible authorization systems, and leverage this knowledge to identify potential topics for future research, as well as a novel method for attacking trust negotiation systems.

## 1 Introduction

Recent research in trust negotiation has been primarily of a theoretical nature, focusing on a number of important issues including languages for expressing resource access policies (e.g., [12,8,18]), protocols and strategies for conducting trust negotiations (e.g., [3,12,13,28]), and logics for reasoning about the outcomes of these negotiations (e.g., [4,27]). These results provide a strong theoretical foundation upon which provably-secure authorization systems can be designed, built, and verified. Some of the techniques discussed in the trust negotiation literature have also been shown to be viable solutions for real-world systems through a series of implementations (such as those presented in [3,9,11,26]) that demonstrate the *feasibility* of using these theoretical advances. However, after several years of research, trust negotiation protocols have yet to make their way into the mainstream.

Prior to deploying access control systems based on trust negotiation, the systems and architectural properties of this technique must be more fully understood. Existing trust negotiation implementations have been developed largely as proofs of concept designed to illustrate the feasibility of the underlying theory and have performed admirably in this capacity. Unfortunately, these proof-of-concept implementations can be difficult to configure and use, and are generally not easily extended or modified. As a result, exploring certain types of systems research problems surrounding trust negotiation becomes difficult. For example:

- Is it possible to unify the myriad formulations of trust negotiation described in the research literature under a common *framework*? Adopting such a framework would make it possible to further deploy and experiment with novel trust negotiation systems and components in a grassroots fashion.
- What are the performance bottlenecks of the trust negotiation *process*, as opposed to those of a specific *implementation*? How can we quantify these costs?
- How can we identify and measure the severity of attacks that are made possible by various approaches to trust negotiation?
- When all other factors are held constant, what are the costs and benefits of using one trust negotiation system component (e.g., negotiation strategy, policy compliance checker, etc.) over another? To what extent do various approaches limit or mitigate attacks?

In an effort to address these types of systems research challenges, we have developed TrustBuilder2, a flexible and reconfigurable Java-based framework for supporting trust negotiation research.<sup>1</sup> TrustBuilder2 supports a plug-in based architecture to allow *any* system component to be modified or replaced by users of the system without requiring modification or recompilation of the underlying framework. TrustBuilder2 is also agnostic with respect to the formats of credentials and policies used during the negotiation. Support for new policy languages, credential formats, or trust negotiation evidence types (e.g., trust tickets [3], uncertified claims [3,4], or proof fragments [27]) can be incorporated by implementing extensions to the TrustBuilder2 data type hierarchy. In this paper, we discuss the design and implementation of TrustBuilder2, as well the results of research carried out using this framework. Specifically, we make the following contributions:

- TrustBuilder2 represents the first fully-configurable framework for trust negotiation. TrustBuilder2 leverages a plug-in based architecture, extensible data type hierarchy, and flexible communication protocol to provide a framework within which numerous trust negotiation protocols and system configurations can be quantitatively analyzed.
- Studies carried out using TrustBuilder2 have identified the primary performance bottlenecks of the trust negotiation process. This has led to the

---

<sup>1</sup> TrustBuilder2 can be downloaded from <http://dais.cs.uiuc.edu/tn>

identification of a novel class of denial of service attacks against trust negotiation systems that differs significantly from the attacks discussed in the research literature.

- TrustBuilder2 demonstrates that adding a high degree of flexibility to advanced authorization frameworks does not necessarily need to incur high overheads. In Section 6, we show that the time spent handling the indirection needed to support user plug-ins and other extensions amounts to less than 0.2% of the total execution time.

The remainder of this paper is organized as follows. In Section 2, we examine previously-developed trust negotiation implementations and discuss the features provided by these systems. In Section 3, we identify a number of useful features that should be provided by frameworks designed to facilitate research on the systems aspects of trust negotiation and the eventual deployment of authorization systems based on trust negotiation; we then identify the subsets of these desiderata that are addressed by existing trust negotiation implementations. Section 4 presents the architecture of the TrustBuilder2 framework for trust negotiation. In Section 5, we explore the ways in which TrustBuilder2 can be extended. Section 6 discusses a performance evaluation of the TrustBuilder2 framework and lessons learned through this process. In Section 7, we examine how TrustBuilder2 addresses the desiderata presented in Section 3. We also discuss attacks on trust negotiation systems, potential research topics uncovered by our performance evaluation, and describe how to obtain the TrustBuilder2 framework. We then present our conclusions in Section 8.

## 2 Background and Related Work

Trust negotiation [25] has been proposed as a potential solution to the recognized problems associated with performing access control in open systems. In trust negotiation, the access policy for a resource is written as a declarative specification of the attributes that an authorized entity must possess to access the resource. In these systems, digital credentials are issued by trusted parties to certify user attributes. For example, a student might have a digital student ID card issued by her university. These credentials are also considered resources, so sensitive credentials can be protected by disclosure policies of their own. In this way, an access request leads to a bilateral and iterative disclosure of credentials and policies between the user and resource provider. Trust is established incrementally, as more and more sensitive credentials are disclosed between the user and resource provider.

Over the last several years, several implementations of trust negotiation systems have been described in the literature. The earliest such implementation was the TrustBuilder architecture for trust negotiation [26]. TrustBuilder is a Java implementation that supports the use of X.509 certificates to encode attributes and XML to represent policies written using the IBM Trust Policy Language (TPL) [8]. The IBM Trust Establishment (TE) compliance checker is used to determine whether a certain set of credentials satisfies a given policy. TrustBuilder

has been embedded into an implementation of TLS [9] and several other protocols to demonstrate the applicability of trust negotiation in existing systems. Unfortunately, TrustBuilder supports the use of only one credential format, one policy language, and one trust negotiation strategy.

Trust- $\mathcal{X}$  [3] is an XML-based framework for supporting trust negotiations in peer-to-peer systems. In Trust- $\mathcal{X}$ , each user creates an  $\mathcal{X}$ -profile that stores  $\mathcal{X}$ -TNL certificates describing their attributes along with uncertified declarations containing information about the user (e.g., preferences, phone numbers, or other such information). To the best of our knowledge, Trust- $\mathcal{X}$  does not support credential formats other than  $\mathcal{X}$ -TNL certificates nor policies specified in any language other than  $\mathcal{X}$ -TNL. To allow users to optimize various aspects of the trust negotiation process, Trust- $\mathcal{X}$  supports a variety of interchangeable trust negotiation strategies. Another particularly innovative feature of the Trust- $\mathcal{X}$  framework is its support for *trust tickets*. Trust tickets are receipts that attest to the fact that a user recently completed some negotiation with another party. These trust tickets can then be presented within some limited lifetime (typically 24-48 hours) to bypass redundant portions of future negotiations with the same party.

In [11], Koshutanski and Massacci describe a trust negotiation framework designed for web services. This framework facilitates the composition of access policies across the constituent pieces of a workflow, the discovery of credentials needed to satisfy these policies, the management of the distributed access control process, and the logic to determine what missing credentials must be located and provided to satisfy a given policy. The use of X.509 and SAML credentials is supported by the framework, as is the use of the negotiation strategies described in [11] and [13]. Policies are represented using a Datalog-based language. To the best of our knowledge, the use of other credential formats, negotiation strategies, or policy languages is not supported.

In [7], De Coi and Olmedilla describe a flexible and expressive trust negotiation implementation. The authors examined the PEERTRUST [20] and PROTUNE [5] systems in an effort to derive a set of common requirements that should be supported by any trust negotiation implementation, and then implemented a framework embodying these requirements. Their system supports PEERTRUST and PROTUNE inference engines, and allows users to add support for other inference engines. Furthermore, users can specify trust negotiation strategies as *action selection algorithms* within their framework. Credentials are expressed as signed logical statements and are loaded from a credential repository that is accessed by their implementation. To the best of our knowledge, the use of other credential formats is not supported.

While not specifically an implementation of a trust negotiation framework, Cassandra [1] is a policy language for distributed access control that supports the specification of policies with a tunable level of expressiveness. The features of Cassandra are such that it can encode a certain, fixed, trust negotiation strategy. A prototype system that uses the Cassandra language has been implemented in OCaml to facilitate research on the features of this policy language.

### 3 System Requirements

Prior to designing the TrustBuilder2 framework for trust negotiation, we first sought to identify the types of features that should be provided by such a framework. To this end, we studied potential uses of trust negotiation in the realms of client/server interactions on the World Wide Web, grid computing, and decentralized information sharing in critical infrastructures. Although space limitations prohibit a full treatment of these use cases,<sup>2</sup> the requirements identified merit discussion since they directed the design of TrustBuilder2. The first set of requirements that we identified relate to the general functionality afforded by the core components of the trust negotiation system.

**Arbitrary Policy Languages.** In many cases, resource providers will wish to be accessible to as many potential clients as possible. To facilitate this, these entities should be able to parse access policies written in a variety of formats (e.g., Cassandra [1], X-TNL [2], TPL [8], RT [18], and XACML [19]). It should be possible to add support for new policy languages to deployed systems easily.

**Arbitrary Credential Formats.** To further enable interactions with a maximal set of users, the system should support the use of multiple credential formats such as X.509 certificates [10] and SAML assertions [6]. It should also be possible to add support for new credential formats to deployed systems easily.

**Interchangeable Negotiation Strategies.** Trust negotiation is by nature a strategy-driven process. Entities should be able to choose negotiation strategies that direct the execution of a trust negotiation session to meet their particular goals (e.g., maximizing privacy or minimizing latency). One can imagine many situations in which the goals of the participants in a negotiation might be conflicting. The use of families of interoperable strategies that allow negotiation participants to choose different, yet compatible, strategies (e.g., as in [28]) should be supported. It should be possible to add support for new negotiation strategies to deployed systems.

**Flexible Policy and Credential Stores.** Clients are likely to utilize several computing devices—such as desktop computers, laptops, PDAs, and smart phones—during the course of their daily activities. It is therefore important that a trust negotiation architecture support interactions with a variety of flexible policy and credential stores (e.g., [21][24]) that will enable users to effectively manage their digital identities across multiple devices.

While these basic flexibility requirements are important, they do not address all aspects of the negotiation process. In particular, we must also consider the ability to add more advanced features that might increase the efficiency, understandability, or functionality of the trust negotiation process.

**Strategy-Driven External Interactions.** Negotiation participants should have the ability to interact with a wide range of external entities that can

---

<sup>2</sup> The complete details of our use case analysis can be found in [14].



help solve difficult problems which may arise during the negotiation. Examples of such interactions might include the calculation of reputations or credential chain discovery. These interactions should be strategy-driven to allow participants to control the amount of time and resources spent pursuing these interactions.

**Advanced Logging Capabilities.** The architecture should include a logging service that can record information regarding any aspect of the negotiation process. Since a high degree of logging is not always needed, the logging subsystem should support the recording of logs at various granularities.

**Tunable Human Involvement.** In some instances, humans may wish to be involved directly in the negotiation process. For example, users may want to specify an “ask me” release policy for a sensitive credential, see a visual representation of the negotiation process for policy evaluation purposes, or be involved in the decision-making process when the negotiation comes to a point where there are multiple execution paths that could be followed rather than relying on a predefined strategy. The framework should support extensions that can add a human “in the loop” if such features are requested.

**Selective Feature Activation.** To enable more efficient or more secure trust negotiation sessions, the features enabled by the framework should be fully configurable. For instance, disabling support for visualization features and external interactions might increase the performance of the system, while disabling third-party plug-ins might increase overall system security and trustworthiness.

**Feature Ordering.** To enhance the performance of the system and its robustness against attack, entities should have the ability to choose the order in which certain functionalities are invoked. For instance, it should be possible for a negotiation strategy to choose the time at which credentials are validated. That is, there may be benefits to delaying credential validation until it is determined that they belong to a minimal satisfying set for some policy, rather than validating them as they are received.

The diversity of use cases that we considered leads us to believe that it represents a useful set of features to support when designing a general-purpose trust negotiation framework. However, the requirements presented above cannot be considered complete, as it is impossible to consider every possible trust negotiation use case. To acknowledge and partially address this gap, we introduce one further requirement that helps ensure additional features can be easily added to the framework.

**Extensibility.** The framework must support the addition of new functionality after deployment without requiring modifications to the existing code base. Example features may include (but are not limited to) the inclusion of new local data processing rules, the enforcement of obligations, and the incorporation of new data types into the negotiation process.

Table 1 identifies the subsets of these requirements addressed by each of the trust negotiation frameworks discussed in Section 2. As shown, no existing trust

**Table 1.** Features supported by existing trust negotiation implementations (Y = yes, N = no, P = partially supported)

	TrustBuilder	Trust- $\mathcal{X}$	Koshutanski	De Coi	Cassandra
Arbitrary policy languages	N	N	N	P	P
Arbitrary credential formats	N	N	P	N	N
Interchangeable negotiation strategies	N	P	P	Y	N
Flexible policy and credential stores	N	N	N	N	N
External interactions	N	N	N	Y	Y
Tunable human involvement	N	N	N	N	N
Advanced logging	N	N	N	P	N
Selective feature activation	N	N	N	N	N
Feature ordering	N	N	N	N	N
Extensibility	N	N	N	P	P

negotiation framework provides even partial support for more than half of the identified features; this is not surprising, given that these implementations were not meant to be general-purpose frameworks.

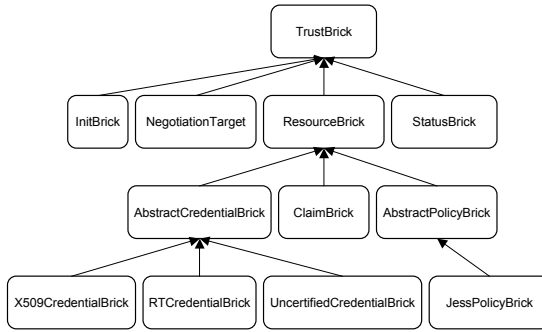
## 4 The TrustBuilder2 Framework

In this section, we describe the design of TrustBuilder2, a Java-based framework for trust negotiation. The primary goal in designing TrustBuilder2 was not to implement one particular trust negotiation protocol, but rather to provide a framework that satisfies the requirements set forth in Section 3, within which any number of trust negotiation techniques can be implemented and evaluated. This led to unique challenges in designing the communication protocol used by negotiation participants, the data type hierarchy used by TrustBuilder2, and the software architecture of the system. In this section, we describe the above facets of the TrustBuilder2 framework.

### 4.1 Communication Protocol and Data Types

One of the first challenges faced when designing TrustBuilder2 was defining a communication protocol that could be interpreted by the framework without constraining the trust negotiation protocols that could be supported. For example, we did not want to mandate that *only* credentials and policies are exchanged during a trust negotiation session, as that would prevent the implementation of protocols such as Trust- $\mathcal{X}$  [3] and PeerAccess [27] within the TrustBuilder2 framework, since these protocols also exchange digitally-signed trust tickets and proof-fragments, respectively. To this end, TrustBuilder2 uses a very simple communication protocol combined with an extensible data type hierarchy to enable the implementation of a wide range of trust negotiation protocols.

*Data type hierarchy.* At a high level, a trust negotiation session is an exchange of messages containing credentials, policies, uncertified claims, and other information between two parties. In order to support the widest possible range of trust negotiation protocols, the core components of the TrustBuilder2 framework (described in Section 4.2) rely heavily on the use of an extensible data



**Fig. 1.** Class hierarchy for several important `TrustBrick` subclasses

type hierarchy. All types of information that might be exchanged between negotiating parties are represented as subclasses of the `TrustBrick` class, which forms the basic building block of the trust negotiation process. In this way, users can extend the data types supported by `TrustBuilder2` without modifying each component in the system; components can simply ignore `TrustBricks` that they do not know how to process, leaving them for other system components to handle. Entities then exchange `TrustMessage` objects containing one or more of these `TrustBricks`.

Figure 1 shows the relationships between several important subclasses of `TrustBrick`. The `InitBrick`, `NegotiationTarget`, and `StatusBrick` classes are used to provide high-level information regarding a negotiation: `InitBricks` are used to establish the parameters of a trust negotiation, a `NegotiationTarget` is used to indicate the particular resource that the initiator of a trust negotiation wishes to access, and a `StatusBrick` may be included in the last message of the negotiation to indicate whether or not the negotiation succeeded in establishing trust between the participants. Any item exchanged during a trust negotiation that could possibly be protected by a release policy is a subclass of `ResourceBrick`. This ensures that `TrustBuilder2` can properly enforce disclosure requirements on data items without necessarily understanding the data item itself.

The `AbstractCredentialBrick` and `AbstractPolicyBrick` classes are used to represent attribute certificates and policies at an abstract level, which enables components of `TrustBuilder2` to handle credentials and policies of various formats without needing to understand the intricacies of each format explicitly. The `X509CredentialBrick` class is used to hold information about X.509 certificates, while the `RTCredentialBrick` class holds information about *RT* credentials [18]. The `UncertifiedCredentialBrick` class provides `TrustBuilder2` with the ability to create “fake” credentials on-the-fly to facilitate the rigorous testing of system components as they are developed. Lastly, the `JessPolicyBrick` class is used to hold policies that can be interpreted by the `CLOUSEAU` compliance checker [15].

In Section 5, we illustrate the ways in which this extensible type hierarchy facilitates the extension of the `TrustBuilder2` framework to incorporate new features, such as support for new policy languages or credential types. Readers

interested in more detail regarding `TrustBrick` or its subclasses should consult the `TrustBuilder2` programmer documentation included with the `TrustBuilder2` distribution.

*The communication protocol.* As previously mentioned, the `TrustBuilder2` communication protocol is nothing more than an exchange of `TrustMessage` objects containing one or more `TrustBricks` between the participants of the negotiation. The first message sent by the initiator of the trust negotiation session contains a single `InitBrick` object describing the `TrustBuilder2` system configurations (i.e., strategy families, credential formats, and policy languages) that she supports, along with other system parameters. If the responder supports a system configuration that is compatible with one of the system configurations proposed by the initiator, he returns a `TrustMessage` containing another `InitBrick` describing this system configuration. At this point, both parties can configure their `TrustBuilder2` framework to use this compatible system configuration during their negotiation session. The initiator then responds with a `TrustMessage` containing a `NegotiationTarget` that indicates the resource that she wishes to access. Beyond this, no constraints are imposed on the contents of these messages; future `TrustMessage` objects exchanged by the participants are handled by the strategy modules (described in the next section) supported by each of the participants, rather than the core `TrustBuilder2` framework. This allows `TrustBuilder2` to support a wide range of trust negotiation protocols without requiring protocol-specific modifications be made to the framework itself.

## 4.2 Software Architecture

Figure 2 presents a high-level architecture diagram of the `TrustBuilder2` runtime system. Note that components enclosed in dashed boxes are not included in the current version of `TrustBuilder2`; they are only meant to serve as example components that could be developed by users as plug-ins and added to the `TrustBuilder2` data path. We now describe each of the major components identified in this diagram and comment on the flow of data between components.

The external interface to the `TrustBuilder2` runtime system is provided by the `TrustBuilder2` class. Trust negotiation sessions are conducted by making a series of calls to methods exposed by this class. When a new trust negotiation session is started, the `TrustBuilder2` class creates and manages a `Session` object that keeps track of all necessary state between rounds of the negotiation. For example, after the exchange of `InitBricks` described in Section 4.1, the `Session` object will contain a description of the `TrustBuilder2` configuration to be used for this session, including the strategy module to use, a list of supported policy languages, and a list of supported credential formats. Any component in the system can add its own internal state to a given `Session` object. This allows components to avoid maintaining this state locally and eases the development of reentrant system components.

During the trust negotiation process, all incoming `TrustMessages` are processed by the `TrustBuilder2` object, which generates a response `TrustMessage` to return

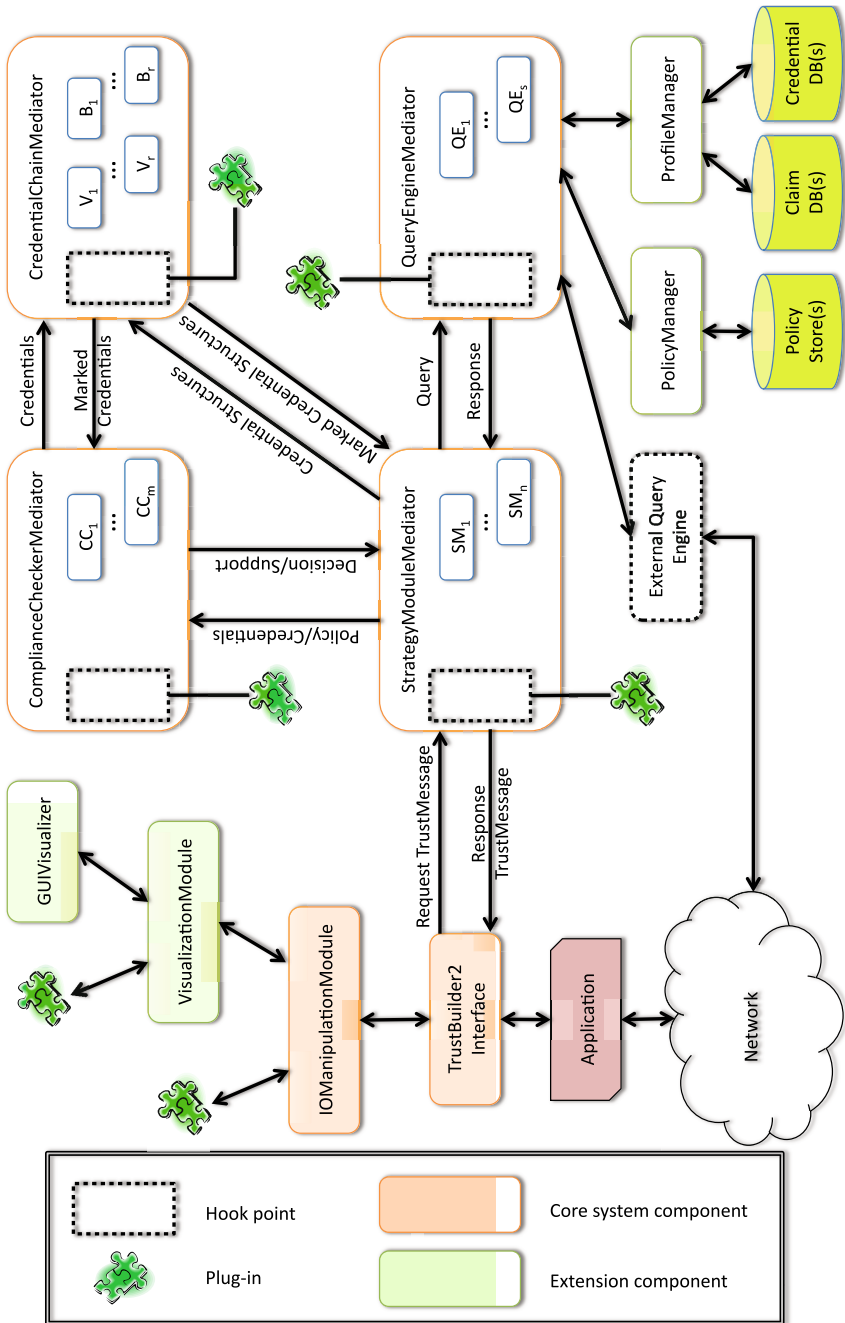


Fig. 2. TrustBuilder2 architecture overview diagram

to the remote participant. Prior to processing a remote `TrustMessage` itself or dispatching it to the `StrategyModuleMediator`, the `TrustBuilder2` object first passes all incoming messages to the `IOManipulationModule`. The `IOManipulationModule` is the first class to process each incoming `TrustMessage` and the last class to process each outgoing `TrustMessage`. This component is capable of loading user-defined plug-ins that can examine and modify all `TrustMessage` objects entering and leaving the `TrustBuilder2` runtime system. The `VisualizationModule` is an example plug-in to the `IOManipulationModule` that provides an interface for writing and using custom logging and visualization components. The `TrustBuilder2` distribution includes two such components: the `GuiVisualizer` class is a plug-in that uses the Swing API to graphically visualize every `TrustMessage` processed by `TrustBuilder2`, while the `BasicConsoleVizualizer` is a plug-in that provides a console logging facility.

The core of the `TrustBuilder2` runtime system—that is, the interfaces to the strategy modules, compliance checkers, credential and policy stores, and credential manipulation routines—is provided by a set of four *mediator* classes. Each mediator class acts as a dispatcher providing access to any number of user-specified trust negotiation system core components. The `StrategyModuleMediator` is responsible for managing the set of installed trust negotiation strategies. Strategies encode the “brains” of a trust negotiation session; given an incoming `TrustMessage` and the existing negotiation state, a strategy responsible for interacting with the `ComplianceCheckerMediator` to analyze policies, the `CredentialChainMediator` to construct and verify credential chains, and the `QueryEngineMediator` to access local trust negotiation evidence or interact with external query services. It then uses the information gleaned from this process to generate a response `TrustMessage` that will be sent to the remote party. Each mediator class provides *hook points* that allow user-developed plug-ins to intercept all calls into the mediator class and all returns from the mediator class. This provides an easy way for users to monitor or modify the flow of information through the `TrustBuilder2` framework.

For the sake of brevity, not every component of `TrustBuilder2` was discussed in this section. Readers desiring a more complete treatment of the components of the `TrustBuilder2` system should consult the programmer documentation available in the `TrustBuilder2` distribution.

### 4.3 Default Configuration and Extensibility

By default, `TrustBuilder2` includes support for a version of the `TrustBuilder1-Relevant` strategy for trust negotiation described in [28] modified to further minimize information disclosure in the event that multiple satisfying sets are found for a given policy during a negotiation. As described above, `TrustBuilder2` supports the use of X.509 V3 credentials during interactions with remote parties but can also use uncertified “test” credentials to exercise the functionality of new plug-ins or components as they are being designed and developed. Plug-ins are provided for the `CredentialChainMediator` that allow `TrustBuilder2` to form a set of credential chains from a collection of credentials of *any* format and to verify

the authenticity of the credential chains that were formed. TrustBuilder2 currently supports the CLOUSEAU compliance checker and can load a user’s policies, credentials, and uncertified claims from repositories on the local file system.

## 5 Case Studies in Extensibility

In this section we discuss the extensibility of TrustBuilder2 at a high level, as well as provide a more detailed treatment of two significant extensions added to the framework after its initial development.

### 5.1 General Extensibility

As was our goal from the outset, almost every component of the TrustBuilder2 framework can either be extended or replaced by a user-defined plug-in. Because the TrustBuilder2 framework was developed using Java, dynamic class loading can be used to incorporate these user plug-ins at runtime without requiring any modification to the TrustBuilder2 framework itself. Extensions to the primary components of TrustBuilder2—that is, the `IManipulationModule`, `StrategyModuleMediator`, `ComplianceCheckerMediator`, `CredentialChainMediator`, and the `QueryEngineMediator`—as well as plug-ins that interpose between these components, can be added the system quite easily. Users simply write and compile plug-ins conforming to the appropriate interfaces and instruct the TrustBuilder2 runtime system to incorporate these modules the next time that a TrustBuilder2 object is created. For instance, adding a new strategy to TrustBuilder2 involves writing a class implementing `StrategyModuleInterface` and adding this class to the list of strategy modules to be loaded by the `StrategyModuleMediator`.

We now discuss how the abstract type hierarchy used by TrustBuilder2 allows support for new credential and policy formats—as well as new forms of negotiation evidence—to be added to the the system without requiring modifications to the underlying framework. As will be shown, this process is very straightforward and allows support for novel trust negotiation features to be easily incorporated into the TrustBuilder2 framework.

### 5.2 X.509 Credentials and Uncertified Claims

Initially, the TrustBuilder2 framework only included support for uncertified “test” credentials, as encoded by the `UncertifiedCredentialBrick` class. These credentials can be easily created and modified and thus allow for rapid and efficient testing of system components. However, to better study the properties of trust negotiation systems that might be deployed in practice, support for more realistic credential types was required. As a result, we added support for uncertified claims encoding user data such as phone numbers or preferences (as in [3,4]), as well as X.509 v3 certificates to the TrustBuilder2 framework.

Supporting uncertified claims required the following two extensions be made to TrustBuilder2. First, the `ClaimBrick` data type was added as a subtype of the

ResourceBrick data type in the TrustBuilder2 type hierarchy (see Figure 11). Subtyping ResourceBrick in this way ensures that uncertified claims can be treated as sensitive and optionally protected by release policies. Second, a loader plug-in was written for the ProfileManager so that uncertified claims could be loaded from the file system. In total, less than 300 lines of commented code had to be written to support the addition of uncertified claims to TrustBuilder2.

Adding support for X.509 v3 certificates to TrustBuilder2 was accomplished in a similar manner. Specifically, the X509CredentialBrick data type was added as a subtype of the AbstractCredentialBrick type and another loader plug-in was written for the ProfileManager so that X.509 certificates could be loaded from the file system. The X509CredentialBrick data type wraps the functionality of the X.509 data type supported by Java natively and provides additional methods that extract attribute information from a credential's extension OID fields, populate the data structures used by AbstractCredentialBrick objects, create and verify proof of ownership challenges, and verify the issuer signatures. Since TrustBuilder2's default policy compliance checker, credential chain construction algorithms, and credential chain verification algorithms operate on AbstractCredentialBrick objects, no further modifications were needed for TrustBuilder2 to support X.509 v3 certificates. Fewer than 1000 lines of commented code were needed to implement the plug-ins required to include this support.

### 5.3 RT Credentials and Policies

To further extend the functionality of TrustBuilder2, we have also implemented support for  $RT_0$  and  $RT_1$  credentials and policies [18]. Adding support for the necessary credential types involved a process similar to that followed for supporting X.509 credentials. That is, TrustBuilder2's type hierarchy was extended to include  $RT$  credentials, and a loader plug-in was written to read these credentials from disk. However, since Java does not support  $RT$  credentials natively, considerably more code had to be written than was the case for adding support for X.509. In total, approximately 3500 lines of commented code were required to add support for loading, parsing, and using these types of credentials within the TrustBuilder2 framework.

In general, adding support for a new policy language to TrustBuilder2 would require developing a new policy compliance checker that is capable of analyzing the satisfaction of this new type of policy. Such a compliance checker would take the form of a plug-in to the ComplianceCheckerMediator. However, this was not the case for  $RT_0$  and  $RT_1$  policies. Recent results [15] show that these types of policies can actually be compiled into a format that can be efficiently analyzed by the CLOUSEAU compliance checker, which is already supported by TrustBuilder2. Currently, policies must be compiled in an offline manner prior to being used by TrustBuilder2, which limits the credential chain discovery functionality supported by  $RT$ . To overcome this barrier, we plan to implement a plug-in that interposes between the StrategyModuleMediator and the ComplianceCheckerMediator and compiles  $RT$  policies at runtime.



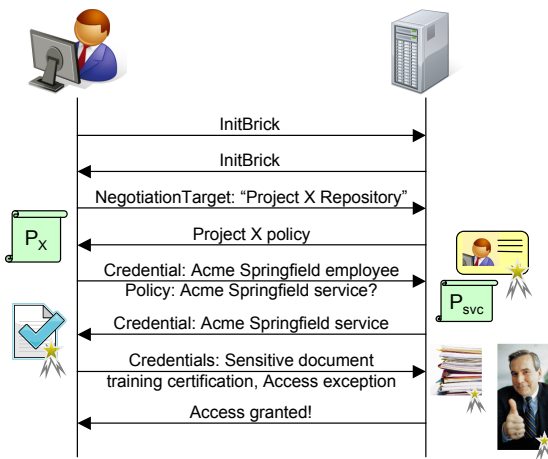
## 6 Performance Evaluation and System Profiling

We now discuss the results of a performance evaluation of the TrustBuilder2 framework. Our primary goals in this investigation were to evaluate the overheads associated with the flexible nature of TrustBuilder2 and to better understand the bottlenecks involved in the trust negotiation process. We then discuss a novel type of denial of service attack and several potential research directions uncovered during this analysis.

### 6.1 The Scenario

Our scenario was designed to mimic a trust negotiation scenario that might take place in one branch (Acme Springfield) of a national-scale corporation (Acme Fabrication). In this scenario, an employee wants to access a file server containing sensitive files related to “Project X.” The policy protecting the Project X file repository states that an authorized entity must be either (i) a full-time employee of Acme Springfield that has an Acme Fabrication issued sensitive document training certification and works in department 2460–2469 or (ii) a full-time employee of Acme Springfield that has an Acme Fabrication issued sensitive document training certification, works in department 2400–2499 and was granted an “access exception” for Project X by either Alice or Bob. This policy was thought to be a reasonable example of a negotiation that one might see in a large corporation as it is much simpler than managing a long access control list, but also includes provisions for the explicit white-listing of people who are not authorized by the blanket policy. Furthermore, entities on the white-list can easily be traced back to the employee authorizing them.

The client in our scenario has a valid employee ID stating that he is a full-time employee in department 2442 of Acme Springfield, a sensitive documents training credential, and an access exception issued by Bob. Figure 3 illustrates



**Fig. 3.** A simplified view of the trust negotiation used during our experiments

this example negotiation scenario graphically. The first two messages exchanged during the negotiation contain configuration information used by TrustBuilder2 to establish the parameters for the negotiation session. The second message sent by the client indicates his interest in accessing the file server associated with Project X. The second message sent by the file server releases the policy protecting this file server to the client. The client can satisfy this policy, but is not willing to disclose his security clearance or access exception unless the server can prove that it is operated by Acme Springfield. As such, the third message sent by the client discloses the release policy protecting these credentials and the credential chain ending with his employee ID. Note that supporting credentials are not shown in Figure 3. At this point, the file server validates the proof-of-ownership associated with the employee's employee ID and accepts this credential. It also then discloses the credential chain that identifies the file service as operated by Acme Springfield. The client verifies this credential chain and the proof-of-ownership associated with the leaf credential in the chain and then discloses his sensitive documents training credential and his access exception to the file server. The file server verifies the proofs-of-ownership associated with these credentials and then grants the client access to the service.

## 6.2 The Experiments

We used the above trust negotiation scenario to conduct two experiments. In the first experiment, a client application made a TCP connection to a server application and carried out the trust negotiation described by Figure 3 using an `ObjectOutputStream` to write `TrustMessages` to the remote server and an `ObjectInputStream` to read response `TrustMessages`. When the negotiation succeeded, the client would disconnect from the server. This entire process was repeated 100 times. The client and server applications were both executed from the system command prompt using JDK 1.5.0\_06. This experiment was designed to enable us to study the average execution time of a trust negotiation session.

In the second experiment, we sought to profile the execution of the TrustBuilder2 runtime system to gain a better understanding of the costs of the various components of a trust negotiation. In this experiment, the server process described above was started from the system command line using JDK 1.5.0\_06. The client application was loaded into the Eclipse development environment and profiled using the Eclipse Test and Performance Tools Platform (TPTP) tracing and profiling tools plug-in version 4.2.1.

In our experiments, the TrustBuilder2 objects used by the client and server processes supported only the use of X.509 credentials encoded as `X509CredentialBrick` objects. All X.509 credentials used during this scenario encoded RSA key pairs. Further, each credential was represented as a unique X.509 certificate with its own key pair. Both the client and server processes supported the use of the CLOUSEAU compliance checker. The strategy used by both parties was the variant of the TrustBuilder1-relevant strategy discussed in Section 4.3 that is implemented by the `MaximumRelevantStrategy` class included in the TrustBuilder2 distribution. Credential chains were built using the `SimpleChainBuilder`

class and verified using the `RootToLeafVerifier` class. The `IOManipulationModule` was disabled at both the client and server. The experiments described above were run using a single machine, rather than two machines, as we were more interested in the computational costs of the trust negotiation than the communication latencies imposed by routing packets through an Ethernet network. The machine that we used had a 3.2 GHz Intel Pentium 4 processor, 1 GB of RAM, and was running Gentoo Linux (kernel 2.6.12).

### 6.3 Results

After conducting the first experiment, we found that the average time to conduct the aforementioned trust negotiation session using TrustBuilder2 was 434.73 ms with a standard deviation of 97.56 ms. This is at least an order of magnitude faster than a trust negotiation session carried out using the original TrustBuilder framework, as a similar negotiation takes seconds on average within that framework [16]. We did find that the first trust negotiation session took roughly three times as long as an average negotiation (1350 ms) due to the cost of Java initially loading the classes used by the TrustBuilder2 framework. We do not see this as a problem, however, as it is likely that TrustBuilder2 objects will be used for multiple negotiations and therefore this initial cost will quickly be amortized, as it was in our experiments.

In our second experiment, we found that the majority of the time spent in one of three tasks: using the compliance checker ( $\approx 49\%$ ), reading from and writing to I/O streams ( $\approx 15.5\%$ ), and signing proof-of-ownership challenges ( $\approx 14.4\%$ ). We also found that the overheads required to support plug-in loading and interposition amount to less than 0.2% of the overall cost of the trust negotiation process. This implies that the flexibility afforded by the TrustBuilder2 framework does not, in and of itself, carry the steep overheads that we had originally anticipated. Of course, loading inefficient or otherwise expensive plug-ins could easily increase the cost of a trust negotiation.

## 7 Discussion

In this section, we revisit the requirements presented in Section 3 and discuss the ways in which they are met by TrustBuilder2. We then discuss potential implications of the performance results obtained in Section 6.

### 7.1 Requirements Redux

In Section 3, we introduced ten requirements that should be provided by frameworks for exploring the systems aspects of trust negotiation. Section 5 illustrated the ways in which plug-in extensions to TrustBuilder2 can be used to meet the *arbitrary policy languages*, *arbitrary credential formats*, *interchangeable negotiation strategies*, *flexible policy and credential stores*, and *extensibility* requirements. The plug-in interface for defining strategy modules does not

place any constraints on how the strategy behaves, which enables user-defined strategy modules to meet the *tunable human involvement* and *feature ordering* requirements. The VisualizationModule plug-in to the IOManipulationModule enables advanced logging and visualization features, thus meeting the *advanced logging capabilities* requirement. Finer-grained logging can be accomplished by placing calls to the logger at the mediator hook points described in Section 4.2. The QueryModuleMediator can be used to allow the TrustBuilder2 framework to interact with processes external to the negotiation at hand simply by developing new query module plug-ins, thereby meeting the *strategy-driven external interactions* requirement. Finally, each of the plug-ins to the TrustBuilder2 system can be individually enabled or disabled, thereby meeting the *selective feature activation* requirement.

## 7.2 Attacks and Future Research

One striking result from the performance evaluation presented in Section 6 is that nearly half of a trust negotiation session is spent interacting with the compliance checker. During our experiments, the client process spent, on average, 226 ms interacting with the compliance checker during a single trust negotiation. The complexity of the compliance checking process has also been observed in other, independent trust negotiation implementations (e.g., see [23]). This suggests that a novel and highly-effective denial of service attack against trust negotiation-enabled services is to force the use of the remote party's compliance checker. An attacker can easily accomplish this by either placing release policies on every credential that might possibly be released to the remote party, or by sending spurious policies that the remote party *thinks* are protecting resources that could advance the state of the negotiation. Such an attack involves little overhead for the attacker, yet can consume arbitrary resources on the host being attacked.

This attack is quite different than the types of denial of service attacks on trust negotiation discussed in the research literature. To date, attacks against trust negotiation systems have focused on examining ways to exploit the credential chain construction and verification processes [17,22]. These attacks leverage the disparity in cost between transmitting a credential chain and verifying that the chain is correctly formed to consume resources on the target system. The higher per-unit cost of policy compliance checking when compared to credential verification implies that attacking the compliance checker used by a trust negotiation system can be at least as damaging as attacking its credential chain verification process. Furthermore, malicious entities combining these two attacks can slow the processes of analyzing both local and remote policies at the host being attacked.

Analyzing the cost breakdown of example trust negotiation scenarios not only led to the identification of this attack strategy, but also helped identify future research directions aiming to better optimize trust negotiation systems. For example, an earlier version of our performance analysis led us to explore alternate formulations of the policy compliance checking problem that would allow for more efficient policy analysis than existing theorem proving approaches. The

result was the CLOUSEAU compliance checker, which leverages an efficient pattern matching approach to greatly outperform existing compliance checkers both asymptotically and in practice [15]. However, the attacks described above occur even when using this optimized compliance checker. An interesting direction of future research could be the development of trust negotiation strategies that can detect the above types of compliance checker abuses and either triage “unproductive” negotiations or seek to limit the use of the compliance checker without compromising the completeness property of the trust negotiation protocol.

## 8 Conclusions

In this paper, we presented TrustBuilder2, a flexible framework for investigating the systems aspects of trust negotiation. TrustBuilder2 supports the dynamic loading of new trust negotiation system components—such as strategy modules, compliance checkers, policy and credential storage devices, and logging and visualization modules—without modification to the underlying framework and features an extensible type hierarchy that allows end-users to easily add support for new credential formats and policy languages. By profiling the performance of TrustBuilder2, we found that the system has a number of desirable properties that make it ideal for researching the systems obstacles to deploying trust negotiation systems in practice. Furthermore, our performance evaluation enabled us to uncover a novel class of attacks against trust negotiation systems and led us to identify promising areas of future trust negotiation systems research.

**Acknowledgments.** This research was supported by the NSF under grants IIS-0331707, CNS-0325951, and CNS-0524695, and by Sandia National Laboratories under grant number DOE SNL 541065.

## References

1. Becker, M.Y., Sewell, P.: Cassandra: Distributed access control policies with tunable expressiveness. In: 5th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY 2004), pp. 159–168 (2004)
2. Bertino, E., Ferrari, E., Squicciarini, A.C.: X -TNL: An XML-based language for trust negotiations. In: 4th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY 2003) (2003)
3. Bertino, E., Ferrari, E., Squicciarini, A.C.: Trust-X: A peer-to-peer framework for trust establishment. *IEEE Transactions on Knowledge and Data Engineering* 16(7), 827–842 (2004)
4. Bonatti, P., Samarati, P.: Regulating service access and information release on the web. In: 7th ACM Conference on Computer and Communications Security, pp. 134–143 (2000)
5. Bonatti, P.A., Olmedilla, D.: Driving and monitoring provisional trust negotiation with metapolicies. In: Proceedings of the Sixth IEEE Workshop on Policies for Distributed Systems and Networks (POLICY 2005), June 2005, pp. 14–23 (2005)

6. Cantor, S., Kemp, J., Philpott, R., Maler, E. (eds.): Assertions and protocols for the OASIS security assertion markup language (SAML V2.0). OASIS Standard (March 2005), <http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf>
7. De Coi, J.L., Olmedilla, D.: A flexible policy-driven trust negotiation model. In: Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology, November 2007, pp. 450–453 (2007)
8. Herzberg, A., Mass, Y., Michaeli, J., Naor, D., Ravid, Y.: Access control meets public key infrastructure, or: assigning roles to strangers. In: IEEE Symposium on Security and Privacy (May 2000)
9. Hess, A., Jacobson, J., Mills, H., Wamsley, R., Seamons, K.E., Smith, B.: Advanced client/server authentication in TLS. In: Network and Distributed Systems Security Symposium (February 2002)
10. Housely, R., Ford, W., Polk, T., Solo, D.: Internet X.509 Public Key Infrastructure Certificate and CRL Profile. IETF Request for Comments RFC-2459 (January 1999)
11. Koshutanski, H., Massacci, F.: Interactive access control for web services. In: 19th IFIP Information Security Conference (SEC), August 2004, pp. 151–166 (2004)
12. Koshutanski, H., Massacci, F.: An interactive trust management and negotiation scheme. In: 2nd International Workshop on Formal Aspects in Security and Trust (FAST), August 2004, pp. 139–152 (2004)
13. Koshutanski, H., Massacci, F.: Interactive credential negotiation for stateful business processes. In: Herrmann, P., Issarny, V., Shiu, S.C.K. (eds.) iTrust 2005. LNCS, vol. 3477, pp. 256–272. Springer, Heidelberg (2005)
14. Lee, A.J.: Towards Practical and Secure Decentralized Attribute-Based Authorization Systems. PhD thesis, University of Illinois at Urbana-Champaign (July 2008)
15. Lee, A.J., Winslett, M.: Towards and efficient and language-agnostic compliance checker for trust negotiation systems. In: 3rd ACM Symposium on Information, Computer, and Communication Security (ASIACCS 2008) (March 2008)
16. Lee, A.J., Winslett, M., Basney, J., Von Welch: The Traust authorization service. ACM Transactions on Information and System Security 11(1) (February 2008)
17. Li, J., Li, N., Wang, X., Yu, T.: Denial of service attacks and defenses in decentralized trust management. In: 2nd International Conference on Security and Privacy in Communication Networks (SecureComm) (August 2006)
18. Li, N., Mitchell, J.: RT: A role-based trust-management framework. In: 3rd DARPA Information Survivability Conference and Exposition (April 2003)
19. Moses, T.: XACML 2.0 Core: eXtensible Access Control Markup Language (XACML) Version 2.0. OASIS Standard (February 2005), [http://docs.oasis-open.org/xacml/2.0/access\\_control-xacml-2.0-core-spec-os.pdf](http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf)
20. Nejdil, W., Olmedilla, D., Winslett, M.: Peertrust: Automated trust negotiation for peers on the semantic web. In: Jonker, W., Petković, M. (eds.) SDM 2004. LNCS, vol. 3178, pp. 118–132. Springer, Heidelberg (2004)
21. Novotny, J., Tuecke, S., Von Welch: An online credential repository for the grid: MyProxy. In: 10th International Symposium on High Performance Distributed Computing (HPDC-10) (August 2001)
22. Ryutov, T., Zhou, L., Neuman, C., Leithead, T., Seamons, K.E.: Adaptive trust negotiation and access control. In: 10th ACM Symposium on Access Control Models and Technologies (June 2005)
23. Smith, B., Seamons, K.E., Jones, M.D.: Responding to policies at runtime in Trust-Builder. In: 5th International Workshop on Policies for Distributed Systems and Networks (POLICY 2004) (June 2004)

24. van der Horst, T.W., Seamons, K.E.: Short paper: Thor — the hybrid online repository. In: 1st IEEE International Conference on Security and Privacy for Emerging Areas in Communications Networks (September 2005)
25. Winsborough, W.H., Seamons, K.E., Jones, V.E.: Automated trust negotiation. In: DARPA Information Survivability Conference and Exposition (January 2000)
26. Winslett, M., Yu, T., Seamons, K.E., Hess, A., Jacobson, J., Jarvis, R., Smith, B., Yu, L.: Negotiating trust on the web. *IEEE Internet Computing* 6(6), 30–37 (2002)
27. Winslett, M., Zhang, C., Bonatti, P.A.: PeerAccess: A logic for distributed authorization. In: 12th ACM Conference on Computer and Communications Security (CCS 2005) (November 2005)
28. Yu, T., Winslett, M., Seamons, K.E.: Supporting structured credentials and sensitive policies through interoperable strategies for automated trust negotiation. *ACM Transactions on Information and System Security* 6(1) (February 2003)

# A Relational Wrapper for RDF Reification

Sunitha Ramanujam<sup>1</sup>, Anubha Gupta<sup>1</sup>, Latifur Khan<sup>1</sup>, Steven Seida<sup>2</sup>,  
and Bhavani Thuraisingham<sup>1</sup>

<sup>1</sup>The University of Texas at Dallas, Richardson TX 75080, U.S.A.

{sxr063200, axg089100, lkhan, bxt043000}@utdallas.edu

<sup>2</sup>Raytheon Company, Garland TX 75042, U.S.A.

steven\_b\_seida@raytheon.com

**Abstract.** The importance of provenance information as a means to trust and validate the authenticity of available data cannot be stressed enough in today's web-enabled world. The abundance of data now accessible due to the Internet explosion brings with it the related issue of determining how much of it is trustworthy. Provenance information, such as who is responsible for the data or how the data came to be, assists in the process of verifying the authenticity of the data. Semantic web technologies such as Resource Description Framework (RDF) include the ability to record such provenance information through the process of reification. RDF's popularity has resulted in a demand for modeling and visualization tools. The work presented in this paper, called R2D, attempts to address this demand by innovatively integrating existing, stable technologies such as relational systems with the newer web technologies such as RDF. The work in this paper extends our earlier work by adding support for the RDF concept of reification. Reification enables the association of a level of trust and confidence with RDF triples, thereby enabling the ranking/validation of the authenticity of the triples. Details of the algorithmic enhancements to the various components of R2D that were made to support RDF reification are presented along with performance graphs for queries executed on a database containing crime records data from a police department.

**Keywords:** Resource Description Framework, Data Provenance, Reification, Data Interoperability.

## 1 Introduction

The extensive growth of the Internet and associated web technologies has catalyzed research into the notion of a "Semantic Web". This notion is envisioned to augment human reasoning and data management abilities with automated access, extraction, and interpretation of web information. Amongst the many methodologies and standards that are being released periodically as part of the Semantic Web initiative is the Resource Description Framework (RDF) [1], a domain-independent data model that enables interoperability between applications that exchange machine-comprehensible information on the Internet. RDF records information in the form of triples, each



consisting of a subject, a predicate, and an object. The predicate is typically a verb and denotes the relationship that exists between the subject and the object. RDF's rapidly increasing popularity as a web content data storage paradigm has necessitated research in the field of visualization tools to inspect and manage data stored using this model. While efforts are ongoing to develop new tools for this purpose, alternate research efforts are underway that focus on integrating benefits and features available in existing methodologies with the advantages offered by the newer web technologies.

R2D, the work presented in this paper, is one such alternative research effort the objective of which is to salvage the time, effort, and resources expended in the development of existing, stable, relational tools by reusing them for RDF data visualization purposes. The advantages of relationalizing RDF stores using applications such as R2D are manifold and include continued leveraging of the knowledge gained by relational database domain experts, reduction of learning curves associated with mastery of new tools, and availability of new technology to resource-constrained small and medium-sized organizations unwilling to invest in expensive tools for fledgling technologies such as RDF [2].

R2D enables the visualization, inspection, and examination of RDF stores using traditional and mature relational tools. The gap between the two paradigms is bridged, through R2D, using a JDBC wrapper that presents, at run-time, a virtual relational version of the RDF store, thereby eliminating the necessity to duplicate and synchronize data. This paper extends the work in [3] by incorporating support for the concept of RDF reification at every stage of R2D's deployment.

Reification is an important RDF concept that provides the ability to make assertions about statements represented by RDF triples. With the increasing number of online resources and sources of information that become available each day, the need to authenticate the available sources becomes essential in order to be able to judge the validity, reliability, and trustworthiness of the information [4]. This authentication is facilitated by augmenting the sources with provenance information, i.e., information describing the origin, derivation, history, custody, or context of a physical or electronic object [5]. RDF Reification, a means of validating a statement/triple based on the trust level of another statement [6], is the solution offered by the WWW consortium for users of RDF stores to record provenance information. Thus, RDF reification is a key component of any application requiring stringent records of the basis/foundation behind every piece of information in the data store. In particular, reification plays a critical role in security-intensive applications where it is imperative to maintain the privacy and ownership of sensitive data. The provenance information captured using reification can be used, in such applications, to monitor and control data access. The contributions of this paper are as follows.

- We propose a mapping scheme for relationalization of RDF Stores. The mapping algorithm extends the algorithm in [3] by including new constructs to handle and process reification information.
- Based on the created map file, we propose a transformation process that generates a normalized, domain-specific virtual relational schema corresponding to the RDF store. The transformation algorithm in [3] is extended to include tables and relationships for reification data.
- We extend the SQL-to-SPARQL translation algorithm in [3] by including the ability to optionally retrieve reification data, when present, through joins.

The organization of the paper is as follows. A brief overview of related research efforts in the relational-to-rdf arena, in either direction, is provided in the following section. R2D mapping preliminaries in terms of the high-level system architecture and mapping constructs are given in section 3 while Section 4 presents detailed descriptions of the various algorithms involved in the mapping process. Section 5 highlights the implementation specifics of the proposed system with sample visualization screenshots and performance graphs for a diverse system with sample visualization screenshots and performance graphs for a diverse range of queries on databases of various sizes. Lastly, Section 6 concludes the paper.

## 2 Related Work

With RDF being the current buzzword in the “Semantic Web” community, research efforts are underway in various aspects of RDF such as RDF-ising relational and legacy database systems, transforming traditional SQL queries into RDF query languages such as RDQL and SPARQL, and optimizing performance of queries issued against RDF data sources. However, the overall concept and objectives of R2D are unique since all research efforts attempt to integrate relational database concepts and Semantic Web concepts from a perspective that is opposite to that considered in our work. The only research with objectives very closely aligned with R2D that we have been able to identify till date is RDF2RDB [7] and differences between the two frameworks are tabulated in Table 1.

**Table 1.** Comparison between RDF2RDB and R2D

RDF2RDB	R2D
Involves data replication resulting in resource wastage and synchronization issues	No data replication/ synchronization issues since relational schema is virtual
Requires presence of ontological information (rdfs:class, rdf:property) for successful mapping	No ontological information required. Mapping discovered through extensive examination of triple patterns
Schema may have unnecessary tables and may not be truly normalized	No unnecessary tables created for to 1:N or N:1 relationships
No details on blank nodes or reification data handling	Meaningful transformations included for blank nodes and reification nodes
No SQL-to-SPARQL transformation	Since relational schema is only virtual, comprehensive SQL-to-SPARQL transformation algorithm is included

The D2RQ project [8], an extensively adopted open source project is another significant player in the RDBMS-RDF mapping arena. The goals of D2RQ are the exact reverse of our goals. They attempt to create a mapping from relational databases to RDF Graphs, and transform RDF queries into corresponding SQL queries, thereby making relational data accessible through RDF applications. Our goal, on the other hand, is to enable RDF triples to be accessed through relational applications. RDF123 [9], an open source translation tool, also uses a mapping concept in the spreadsheet domain where the users define mappings between the spreadsheet semantics and RDF graphs for richer translation.

Other efforts in the reverse direction include [10] where Perez and Conrad use relational.OWL to extract the semantics of a relational database and automatically transform them into a machine-readable and understandable RDF/OWL ontology. A few contributions that actually consider the mapping process from the same perspective as our research (i.e., from RDF to relational model) are the ones listed in [11]. However, all models are very generic, involving non-application-specific tables such as resources, literals, statements etc. that would make the determination of the problem domain addressed by the model difficult without examining the actual data. Further, none of the models discuss the concept of RDF reification and the relational transformation of the same. In contrast, R2D details a mapping scheme for representing provenance information in a relational format and enables the users to actually arrive at a complete Entity-Relationship Diagram.

The query processing component of R2D which comprises the SQL-to-SPARQL transformation process, once again, has no comparable counterpart while many efforts, [12, 13, 14], are underway in the other direction, namely, SPARQL-to-SQL conversion. Chebotko, et. al. [12] propose an algorithm to translate SPARQL queries with arbitrary complex optional patterns to an equivalent SQL statement. Chen, et. al. [13] discuss a methodology that supports integration of heterogeneous relational databases using the RDF model. An SQL-based RDF Querying Scheme is presented in [14] where the RDF querying capability is made a part of the SQL. The current research efforts presented above indicate that no current solutions address the issue of enabling relational applications to access RDF data without data replication. Hence, to the best of our knowledge, R2D is unprecedented.

### 3 R2D Architecture and Preliminaries

Figure 1 illustrates the architecture of the proposed system along with the specific R2D modules that are responsible for each function provided by R2D. R2D's primary objective is to present, through a JDBC interface, a relational equivalent of RDF

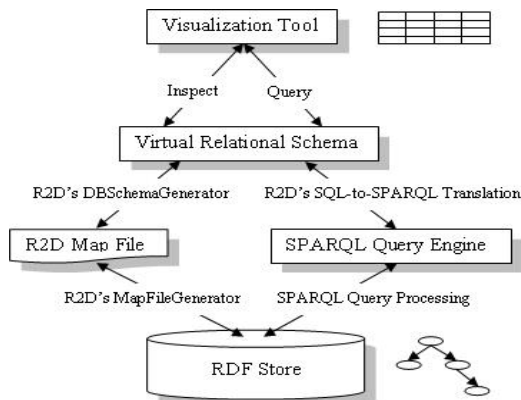


Fig. 1. R2D System Architecture and Modules

triples stores to visualization tools that are based on a relational model. It also provides an SQL Interface that generates SPARQL versions of SQL queries and passes the same to the SPARQL Query Engine layer for processing and RDF data retrieval.

At the heart of the RDF-to-Relational transformation process is the R2D mapping language – a declarative language that expresses the mappings between RDF Graph constructs and relational database constructs. In order to better understand the constructs comprising the R2D mapping language, let us consider the sample scenario in Figure 2.

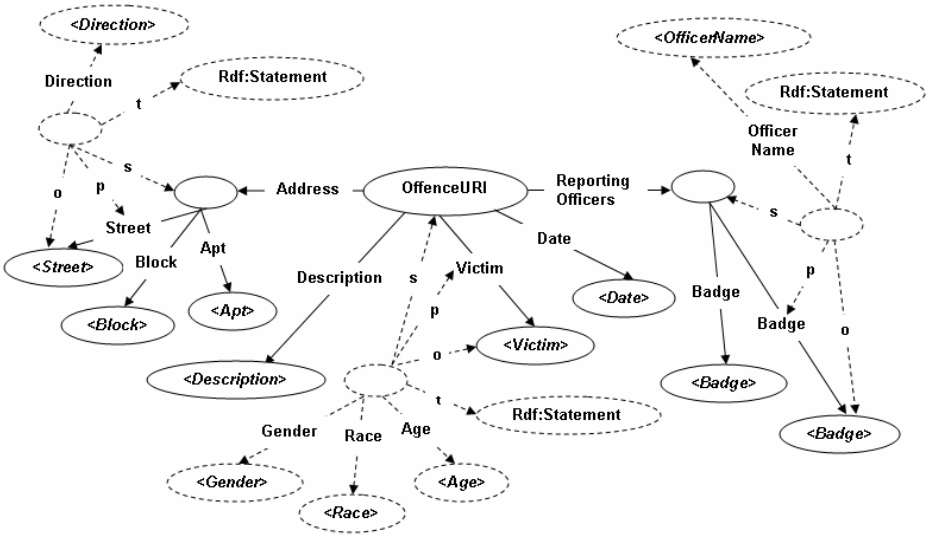


Fig. 2. Sample Scenario involving Crime Data

Every solid node with outgoing edges, such as *OffenceURI*, represent a subject/resource. Edges, such as *Address*, *Description*, and *Victim*, represent predicates and the solid nodes at the end of the edges, such as *<Street>*, *<Description>*, and *<Victim>*, represent objects. Empty solid nodes, such as the nodes at which the *Address* and *ReportingOfficer* predicates terminate, represent blank nodes. The nodes in dashed lines represent reified nodes with the “s”, “p”, “o”, and “t” representing the “rdf:subject”, rdf:predicate, “rdf:object”, and the “rdf:type” predicates of the reification quad. Other predicates of the reification nodes (other than “s”, “p”, “o”, and “t” predicates) represent non-quad predicates. The non-quad reification properties chosen in this example may not represent actual provenance information. They were primarily chosen to illustrate proof of concept. Elements of Figure 2 are used, wherever applicable, to facilitate better comprehension of the mapping constructs which are discussed in the remainder of the section. Table 2 lists the mappings between some key OWL/RDFS ontology terminologies and RDF concepts to appropriate R2D constructs and their relational equivalents.

**Table 2.** Notional Mapping between OWL/RDFS concepts, R2D constructs, and Relational concepts

OWL/RDFS/RDF concepts	R2D Mapping Constructs	Relational Equivalent
rdfs:class	r2d:TableMap	Table
rdf:property	r2d:ColumnBridge	Column
rdfs:domain		Table that the rdf:Property is a column of
rdfs:range	r2d:Datatype	Datatype of the column
rdf:type predicate	r2d:KeyField	Values of the primary key column of the table
Blank Node	r2d:SimpleLiteralBlankNode	Columns in parent table
	r2d:ComplexLiteralBlankNode	Columns in a new join table (symbolizing N:M relationship)
	r2d:{Simple/Complex} resource-BlankNode	Depending on cardinality, either columns in the parent table (1:N relationship) or columns in a new join table (N:M relationship)
Reification	r2d:ReificationNode	Columns in either the parent table or in a new join table

The constructs listed in Table 2 above are described in more detail below along with some of the R2D mapping constructs pertaining to regular resources and blank nodes that are essential in order to effortlessly comprehend the work in this paper. A complete list of mapping constructs can be found in [3].

**r2d:TableMap:** The r2d:TableMap construct refers to a table in a relational database. In most cases, each rdfs:class object will map to a distinct r2d:TableMap, and, in the absence of rdfs:class objects, the r2d:TableMaps are inferred from the instance data in the RDF Store. Typically, every solid node with multiple predicates in an RDF graph maps into an r2d:TableMap if a similar TableMap does not already exist.

*Example:* The RDF graph in Figure 2 results in the creation of a TableMap called “*Offence*”.

**r2d:ColumnBridge:** r2d:ColumnBridges relate single-valued RDF Graph predicates to relational database columns. Each rdf:Property object maps to a distinct column attached to the table specified in the rdfs:domain predicate. In the absence of rdf:property/domain information, they are discovered by exploration of the RDF Store data.

*Example:* The *Description*, *Victim*, and *Date* predicates in Figure 2 become r2d:ColumnBridges belonging to the *Offence* r2d:TableMap.

**r2d:SimpleLiteralBlankNode:** r2d:SimpleLiteralBlankNodes help relate RDF Graph blank nodes that consist purely of distinct simple literal objects to relational database columns. Predicates off of an r2d:SimpleLiteralBlankNode become columns in the table corresponding to the subject of the blank node.

*Example:* The object of the *Address* predicate in Figure 2 is an example of an r2d:SimpleLiteralBlankNode which has distinct literal predicates of *Street*, *Block*, and

*Apt*, which are, in turn, translated into columns of the same names in the *Offence* r2d:TableMap.

**r2d:ComplexLiteralBlankNode:** This construct refers to blank nodes in an RDF Graph that have multiple object values for the same subject and predicate concept associated with the blank node. An r2d:ComplexLiteralBlankNode results in the generation of a separate r2d:TableMap with a foreign key relationship to the table representing the subject resource of the blank node.

*Example:* The object of the *ReportingOfficers* predicate in Figure 2 is an example of an r2d:ComplexLiteralBlankNode that has multiple object (*Badge*) values for the subject (*OffenceURI*) and predicate (*ReportingOfficers*) concept associated with the blank node. The relational transformation for *ReportingOfficers* involves the generation of an r2d:TableMap of the same name. This *ReportingOfficers* r2d:TableMap includes as columns a *Type* field that holds the values of the predicates off of the CLBN (in our sample scenario, the *Type* field will hold a value of “*Badge*”), and a *Value* field that holds the object values of the predicates off of the CLBN. Additionally, the r2d:TableMap also includes, as foreign key, the *Offence\_PK* column which references the primary key of the *Offence* r2d:TableMap.

The concept of reification is supported using many of these previously defined constructs along with a few new constructs that are described below.

**r2d:ReificationNode:** The r2d:ReificationNode construct is used to map blank nodes associated with “reification quads”. Under certain scenarios an r2d:ReificationNode results in the generated of a new “reification” r2d:TableMap. These scenarios are discussed in detail in Section 4.2. The mapping constructs specific to r2d:ReificationNodes are discussed next.

*Example:* The non-solid nodes corresponding to the *Address-Street* predicate, the *Victim* predicate, and the *ReportingOfficers-Badge* predicate in Figure 2 are examples of r2d:ReificationNodes named *Address\_Street\_Reif*, *Victim\_Reif*, and *ReportingOfficers\_Badge\_Reif* respectively.

**r2d:BelongsToTableMap:** This construct connects an r2d:ReificationNode to the r2d:TableMap corresponding to the resource associated with “rdf:subject” of the r2d:ReificationNode. This information is recorded in the R2D Map File for use during the SQL-to-SPARQL translation.

*Example:* *OffenceURI* is the value of the *rdf:subject* predicate of the *Victim\_Reif* r2d:ReificationNode. The r2d:TableMap corresponding to *OffenceURI* is *Offence*. Hence, the r2d:BelongsToTableMap construct corresponding to *Victim\_Reif* is set to a value of *Offence*, thereby connecting the reification node to a relational table.

**r2d:BelongsToBlankNode:** This construct connects an r2d:ReificationNode to the r2d:[Simple/Complex][Literal/Resource]BlankNode corresponding to the blank node associated with the “rdf:subject” of the r2d:ReificationNode.

*Example:* The *rdf:subject* of the *Address\_Street\_Reif* reification node in Figure 2 consists of a blank node resource called *Address*, which is an r2d:SimpleLiteralBlankNode. Hence, for this reification node the r2d:BelongsToBlankNode construct is used to associate *Address\_Street\_Reif* to the *Address* blank node.

**NOTE:** Since the *rdf:subject* of a reification node can either refer to a proper resource or a blank node, the *r2d:BelongsToTableMap* and *r2d:BelongsToBlankNode* constructs are mutually exclusive. These are primarily required to enable the generation of appropriate SPARQL WHERE clauses during SQL-to-SPARQL translation.

**r2d:ReifiedPredicate:** This construct is used to record the predicate corresponding to the “*rdf:predicate*” property of the reification quad mapped by the *r2d:ReificationNode* construct. This information is, again, required for appropriate SPARQL query generation.

*Example:* The complete URI of the *Victim* predicate of *OffenceURI* is recorded under the *Victim\_Reif* reification node using the *r2d:ReifiedPredicate* construct.

Predicates of *r2d:ReificationNodes* are mapped using the *r2d:ColumnBridge* construct described earlier in this section. Some of the important mapping constructs specific to *r2d:ColumnBridges* include:

**r2d:BelongsToReificationNode:** This construct connects an *r2d:ColumnBridge* to an *r2d:ReificationNode* entity and is a mandatory component of *r2d:ColumnBridges* belonging to reification nodes.

*Example:* The *r2d:BelongsToReificationNode* associated with the *Victim\_Gender* *r2d:ColumnBridge* is assigned a value of *Victim\_Reif*, thereby linking the *Victim\_Gender* column with its reification node.

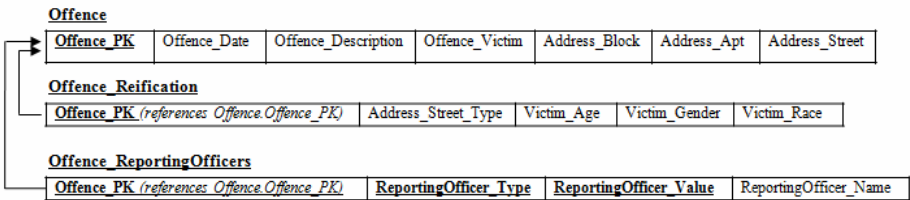
**r2d:DataType:** This construct specifies the datatype of the *r2d:ColumnBridge* to which it is associated and comes into play when the structure of the virtual relational database schema objects is examined.

*Example:* The *Address\_Block* column bridge may have an *r2d:DataType* of *Integer* while the *Victim\_Gender* column bridge has an *r2d:DataType* of *String*.

**r2d:Predicate:** This construct is used to store the fully qualified property name of the predicate which is associated with the reification *r2d:ColumnBridge*. This information is used during the SQL-to-SPARQL translation to generate the SPARQL WHERE clauses required to obtain the value of the *r2d:ColumnBridge*

*Example:* The complete URI of the *Victim\_Gender* predicate of the *Victim\_Reif* reification node is recorded using the *r2d:Predicate* construct.

Figure 3 illustrates the relational schema that is inferred using the above mapping constructs.



**Fig. 3.** Equivalent Relational Schema for Sample Scenario involving Crime Data

The following sections describe how each of the above mentioned R2D constructs is utilized to transform provenance information available in RDF stores through the reification concept into their relational equivalents.

## 4 Reification within the R2D Framework

In order to bring to fruition R2D's vision and objectives, various algorithms were designed and developed to implement each component, highlighted in Figure 1, within the R2D framework. The algorithmic details of each R2D module for translation of regular resources and blank nodes are described in depth in [3] and are omitted from this paper due to space constraints. The following sections discuss the algorithmic aspects specifically associated with the presentation of a relational view of RDF **reification** data.

### 4.1 Mapping Reification Nodes – RDFMapFileGenerator

The RDFMapFileGenerator is the first component in the R2D transformation framework. It is responsible for the generation of a map file containing the correlations between meta-data gleaned from the input RDF store and their relational schema equivalent.

The reification data processing component of the RDFMapFileGenerator is quite straightforward. Every blank node corresponding to a “reification quad” is mapped using the `r2d:ReificationNode` construct. If the “`rdf:subject`” property of the “reification quad” mapped by the `r2d:ReificationNode` construct is a resource, the `r2d:BelongsToTableMap` construct is used to associate the “reification quad” with the `r2d:TableMap` corresponding to the resource. If the “`rdf:subject`” property is a blank node, the `r2d:BelongsToBlankNode` construct is used to associate the “reification quad” to the `r2d:[Simple/Complex][Literal/Resource]BlankNode` associated with the “`rdf:subject`” blank node. Further, if the `rdf:object` property of the “reification quad” refers to another resource, then `r2d:RefersToTableMap` construct is used to store this relationship. This information is used in the case of 1:N relationships between two `TableMap` entities during the SQL-to-SPARQL transformation. Column 1 of Table 3 is the mapping file excerpt for the *Victim\_Reif* and the *Address\_Street\_Reif* reification nodes from Figure 2.

Every non-quad predicate of the reification blank node is mapped using the `r2d:ColumnBridge` construct and is associated with its reification node using the `r2d:BelongsToReificationNode` construct. Furthermore, the datatype of the object corresponding to the non-quad predicate is mapped using the `r2d:Datatype` construct and the URI of the non-quad predicate itself is recorded using the `r2d:Predicate` construct, for use during the SQL-to-SPARQL transformation. An excerpt from the mapping file that includes information for the *Victim\_Gender* and the *Address\_Street\_Direction* properties of the corresponding reification nodes from Figure 2 is listed in Column 2 of Table 3.



**Table 3.** Mapping of Reification Nodes and their Predicates in the R2D Map File

Map File Excerpt for Reification Nodes	Map File Excerpt for Predicates of Reification Nodes
<pre>map:Victim_Reif a r2d:ReificationNode; r2d:belongsToTableMap map:Offence; r2d:datatype xsd:String; r2d:reifiedPredicate &lt;http://Victim&gt;; . map: Address_Street_Reif a   r2d:ReificationNode; r2d:belongsToBlankNode map: Address; r2d:datatype xsd:String; r2d:reifiedPredicate &lt;http://Address/Street&gt;; .</pre>	<pre>map:Victim_Gender a r2d:ColumnBridge; r2d:belongsToReificationNode map:Victim_Reif; r2d:datatype xsd:String; r2d:predicate &lt;http://Reification/Gender&gt;; . map: Address_Street_Direction a r2d:ColumnBridge; r2d:belongsToReificationNode   map:Address_Street_Reif; r2d:datatype xsd:String; r2d:predicate &lt;http://Reification/StreetDirection&gt;; .</pre>

Complex reification nodes, such as ones that contain one or more blank node predicates, are processed using the Depth-First-Search tree algorithm (similar to mixed blank nodes processing [3]). Every blank node encountered during DFS is mapped using the `r2d:SimpleLiteralBlankNode` construct. Every predicate of the blank node is mapped using the `r2d:ColumnBridge` construct and is linked to its parent blank node using the `r2d:BelongsToBlankNode` construct. Every complex reification node is mapped using the `r2d:ComplexReificationNode` construct. Blank node objects belonging to an `r2d:ComplexReificationNode` are connected to the `r2d:ComplexReificationNode` using the `r2d:BelongsToReificationNode` construct.

#### 4.2 Relationalizing Reification Data – DBSchemaGenerator

The second stage of the R2D transformation framework, the `DBSchemaGenerator`, involves the actual virtual, normalized, relational schema generation for the input RDF store based on information in the map file created in stage one. Details of the algorithm pertaining to the relational transformation of reification data are discussed below.

**Case (a):** For every `r2d:TableMap` in the virtual relational schema corresponding to an RDF store an additional `r2d:TableMap` (i.e., a virtual relational table) of type “ReificationTable” is created in the schema if any of the following conditions hold:

- a) An `r2d:ColumnBridge` corresponding to a predicate of a resource that maps to the `r2d:TableMap` is reified
- b) A `r2d:MultiValuedColumnBridge` (MVCB) that results in the addition of a column to this `r2d:TableMap` is reified
- c) A predicate corresponding to an `r2d:SimpleLiteralBlankNode` (SLBN) associated with a resource that maps to the `r2d:TableMap` is reified
- d) An `r2d:ColumnBridge` associated with a predicate of an `r2d:SimpleLiteralBlankNode` (SLBN) object is reified.

This additional reification table houses the columns corresponding to every single-valued property (other than the 4 properties comprising the quad) of the “reification quads” arising from the 4 conditions described above. In order to better understand the intricacies of the algorithm let us consider the scenario depicted in Figure 2.

The reification of the *Victim* predicate in Figure 2 is an example of condition (a) above while reification of the *Street* predicate of the *Address* SLBN is an example of condition (d). The relational transformation of these reification nodes results in the creation of a new virtual relational table (called *Offence\_Reification*) with the following columns (corresponding to the predicates of the reification quads): *Address\_Street\_Direction*, *Victim\_Gender*, *Victim\_Race*, and *Victim\_Age*.

**Case (b):** In the case of reification of MultiValuedColumnBridges that result in the creation of a new join table and reification of other kinds of blank nodes other than SLBNs (more details on the various blank node types and their relational representations can be found in [3]), no new reification table is created. Non-quad properties corresponding to such reifications are added as columns to the existing *r2d:TableMaps* resulting from relationalization of the MVCBs and blank nodes. Reification of the *Badge* predicate of the ComplexLiteralBlankNode (CLBN) *ReportingOfficers* in Figure 2 is one such example where an *OfficerName* column (corresponding to the non-quad predicate of the reification node for *Badge*) is added to the *Offence\_ReportingOfficers* TableMap that results from the relational transformation of the *ReportingOfficers* CLBN.

Complex reification nodes are nodes where non-quad predicates include one or more (nested) blank nodes. Due to the numerous types of such mixed combinations that are possible, it would be nearly impossible to arrive at an accurate normalized representation of the same. Hence, *r2d:ComplexReificationNodes* are processed by flattening their relational equivalents. Depending on whether Case (a) or Case (b) is applicable to the *r2d:ComplexReificationNode*, either a new or an existing table houses the reification columns. Predicates of literal and resource objects that are at the leaf nodes of the tree rooted at the *r2d:ComplexReificationNode* are translated into columns in that table.

### 4.3 Querying Reification Data – SQL-to-SPARQL Translation

The final stage of the R2D transformation framework involves the translation of SQL statements issued against the virtual relational schema generated by stage 2 into equivalent SPARQL queries that are executed against the actual RDF store. This is achieved through the translation algorithm, which also ensures that triples retrieved from the RDF store are returned to the relational visualization tool in the expected tabular format. The translation algorithm presented here extends the earlier version [3] by including the ability to translate queries issued against the virtual tables corresponding to **reification** data.

The SQL-to-SPARQL translation process transforms single or multiple table queries with or without multiple where clauses (connected by AND, OR, or NOT operators) and Group By clauses. Due to space constraints, only a high level description of the algorithm is discussed below along with examples to illustrate the translation process.

In order to understand the intricacies of the translation algorithm, let us consider the following SQL query based on the scenario depicted in Figure 2.

```
SELECT address_street, address_street_direction, address_block, victim_gender,
reportingOfficers_badge, reportingOfficers_name FROM Offence, Offence_Reification,
Offence_ReportingOfficers where Offence.Offence_pk = Offence_Reification.Offence_pk AND
Offence.Offence_pk = Offence_ReportingOfficers.Offence_pk WHERE address_block = '1100';
```

The first step in the translation process involves the generation of the SPARQL SELECT clause. For every field in the original SQL SELECT list, a variable is added to the SPARQL SELECT list. The SPARQL SELECT list after fields processing is:

```
SPARQLSelect = SELECT ?address_street, ?address_street_direction, ?address_block, ,
?victim_gender, ?reportingOfficers_badge, ?reportingOfficers_badge_name
```

The processing of regular columns for generation of SPARQL WHERE and FILTER clauses is described in [3]. The resulting SPARQL WHERE clause after processing of regular, non-reification columns as detailed in [3] is as follows:

```
SPARQLWhere = WHERE {
  ?Offence <http://Offence/Address> ?Offence_Address .
  ?Offence_Address <http://Offence/Address/Street> ? address_street .
  ?Offence_Address <http://Offence/Address/Block> ? address_block .
  ?Offence <http://Offence/ReportingOfficers> ?Offence_ReportingOfficers .
  ?Offence_ReportingOfficers http://Offence/ReportingOfficers/Badge ?reportingOfficers_badge
  FILTER (?address_block = '1100' ) }
```

(a) For fields belonging to tables of type “ReificationTable” corresponding to non-complex reification nodes, if the reification quad to which the field belongs reifies a resource (and not a blank node), clauses of the form *[OPTIONAL] { ?reificationQuad <rdf:subject> ?resourceTableMap . ?reificationQuad <rdf:predicate> ?reificationQuad.r2d:ReifiedPredicate . ?reificationQuad <non-quadPredicate> ?reificationColumn . ?reificationQuad <rdf:object> ?reifiedObjectField .}* are added to the SPARQL WHERE clause. The reification quad corresponding to the *victim\_gender* column is one such reification. The *OPTIONAL* keyword is optional and is only required for queries involving outer joins. Also, if the field corresponding to the object being reified is not part of the SPARQL WHERE clause, an appropriate selection clause is added to the same. The SPARQL WHERE clauses resulting from the processing of the *victim\_gender* column are:

```
REIFClause1 = ?Offence <http://Offence/Victim> >offence_victim .
?Victim_Reif <rdf:subject> ?Offence . ?Victim_Reif <rdf:Predicate>
<http://Offence/Victim> . ?Victim_Reif <rdf:Object> ?offence_victim . ?Victim_Reif
<http://Offence/Victim/Gender> ?victim_gender.
```

Processing of reification columns belonging to {Literal/Resource} MultiValuedColumnBridge ({L/R}MVCB) tables is similar to the above case with an additional step to identify the parent table from which the {L/R}MVCB table is derived through normalization.

In the case of RMVCB tables where the *rdf:object* of the reification quad is a resource that maps to another *r2d:TableMap* (through the *r2d:refersToTableMap* construct), an additional clause of the form

```
?subjectResourceTableMap <reificationQuad.r2d:ReifiedPredicate> ?objectResourceTableMap .
```

is added to the SPARQL WHERE clause.

(b) For fields belonging to tables of type “ReificationTable”, if the reification quad to which the field belongs reifies a blank node, clauses of the form given below are added to the SPARQL WHERE clause. Further, if the *rdf:object* of the reification quad is a resource mapping to another *r2d:TableMap* then the following additional

clause of the form *?BlankNode* <reificationQuad.r2d:ReifiedPredicate> *?objectResourceTableMap* . is appended to the SPARQL WHERE Clause.

*?ParentTableofBlankNode* <BlankNodePredicate> *?BlankNode* . [OPTIONAL] {*?reificationQuad* <rdf:subject> *?BlankNode* . *?reificationQuad* <rdf:predicate> *?reificationQuad.r2d:ReifiedPredicate* . {*?reificationQuad* <rdf:object> *?reifiedObjectField* .*?reificationQuad* <non-quadPredicate> *?reificationColumn*}

The *address\_street\_direction* reification column belonging to the “Address” SLBN in Figure 2 is an example such a reification and the addition to the SPARQL WHERE clause after processing of the same is as given below.

*REIFClause2* = *?Address\_Street\_Reif* <rdf:subject> *?Offence\_Address* . *?Address\_Street\_Reif* <rdf:Predicate> <<http://Offence/Address/Street>> . *?Offence\_Address* <rdf:Object> *?address\_street* . *?Address\_Street\_Reif* <<http://Offence/Address/Street/Direction>> *?address\_street\_direction* .

Reification columns belonging to CLBNs are processed in a manner very similar to the previous scenario (Scenario (b)). The reification column *ReportingOfficers\_Badge\_Name* belonging to the “ReportingOfficers” CLBN in Figure 2 falls in this category and the SPARQL WHERE clauses for this reification are as follows.

*REIFClause3* = *?ReportingOfficers\_Reif* <rdf:subject> *?Offence\_ReportingOfficers* . *?ReportingOfficers\_Reif* <rdf:Predicate> <<http://Offence/ReportingOfficers/Badge>> . *?ReportingOfficers\_Reif* <rdf:Object> *?reportingOfficers\_badge* . *?ReportingOfficers\_Reif* <<http://Offence/ReportingOfficers/Badge/Name>> *?reportingOfficers\_badge\_name* .

Reification columns belonging to r2d:TableMaps corresponding to all other kinds of blank nodes are processed using either scenario (a) or (b) depending on the whether the “rdf:subject” of the reification node is a resource or a blank node.

(c) For fields derived from complex reification nodes, the sequence of predicates leading from the reification node to the (leaf) field are obtained by traversing the tree structure stored during the map file generation process. A WHERE clause is added to the SPARQL WHERE for each of the predicates in sequence.

After the translation procedures described above are applied to the given example SQL statement, the final transformed SPARQL Query is:

*SPARQL Statement* = *SPARQLSelect* + *SPARQLWhere* + *REIFClause1* + *REIFClause2* + *REIFClause3*

The transformed SPARQL Query is executed and the retrieved data is returned in relational format seamlessly.

## 5 Experimental Results

The hardware used for our simulation exercises was a Windows machine with 4GB RAM and 2 GHz Intel Dual Core processor. The software platforms and tools used include Jena 2.5.6 to manipulate the RDF triples data, MySQL 5.0 to house the RDF data in a persistent manner, and DataVision v1.2.0, an open source relational tool, [<http://datavision.sourceforge.net/>], to visualize, query, and generate reports based on the RDF data. Lastly, BEA Workshop Studio 1.1 Development Environment along

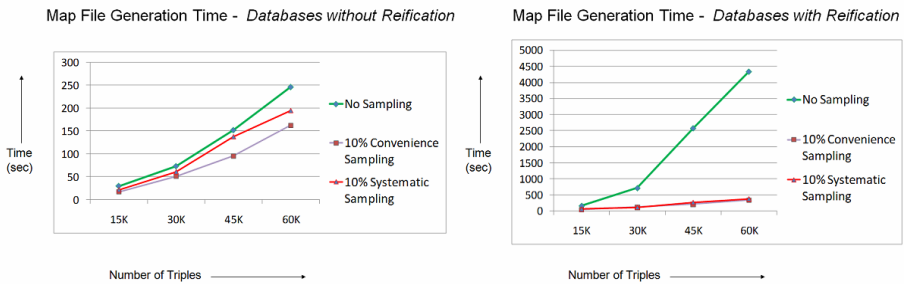
with Java 1.5 was used for the development of the algorithms and procedures detailed in Section 4.

## 5.1 Experimental Datasets

The dataset used in the experiments below is a subset of crime data downloaded from a police department website. The data has triples pertaining to cities and zip codes where crimes were committed, and details of committed crimes as illustrated in Figure 2. While the DataVision screenshots include actual, valid crime data, the voluminous datasets used in the query performance evaluations was artificially generated through a data loading program. However, the structure of the simulated data was kept identical to that of the actual crime dataset and, hence, the results obtained can be directly applied to actual crime data of those volumes. For query performance experiments, Jena's in-memory model was used to load and query the data.

## 5.2 Simulation Results

The relational equivalent of the crime data was generated using the algorithms detailed in Sections 4.1 and 4.2. The time taken by the map file generation process without any data sampling incorporated for RDF stores of various sizes, with and without reification information, was compared with time taken for the same process when two sampling methods were applied and the results are illustrated in Figure 4. Reified versions of the crime dataset were created by adding reification information to the *Address (Address\_Type)* and *Victim (Gender, Race, Age)* objects in Figure 2. This reification information was created for 50% of the offence data in the data stores.



**Fig. 4.** Map File Generation Times with/without Sampling for reified/un-reified data

The process is especially time-intensive for large databases without structural information (which is the case with our experimental data set) but this is only to be expected since the `RDFMapFileGenerator` has to explore every resource to ensure that no property is left unprocessed. Furthermore, since even adding reification information for only 50% of the triples in the RDF store resulted in a 25% increase in the size of the data store, the increase in map file generation time for databases with reification information is also predictable. However, the sampling techniques applied improved the performance of the algorithm by a large factor.

Figure 5 is a screenshot of DataVision’s Report Designer along with an inset of the database schema as seen by DataVision. The r2d:SimpleLiteralBlankNode associated with *Offence-Address* is resolved into columns belonging to the *Offence* table, and the r2d:ComplexLiteralBlankNode associated with *Offence-ReportingOfficers* is resolved into a 1:N table of the same name. Reification columns are segregated into corresponding reification tables. This schema is populated through the GetDatabase-MetaData Interface in the Connection class of the JDBC API within which the two algorithms, RDFMapFileGenerator and DBSchemaGenerator, are triggered. At this

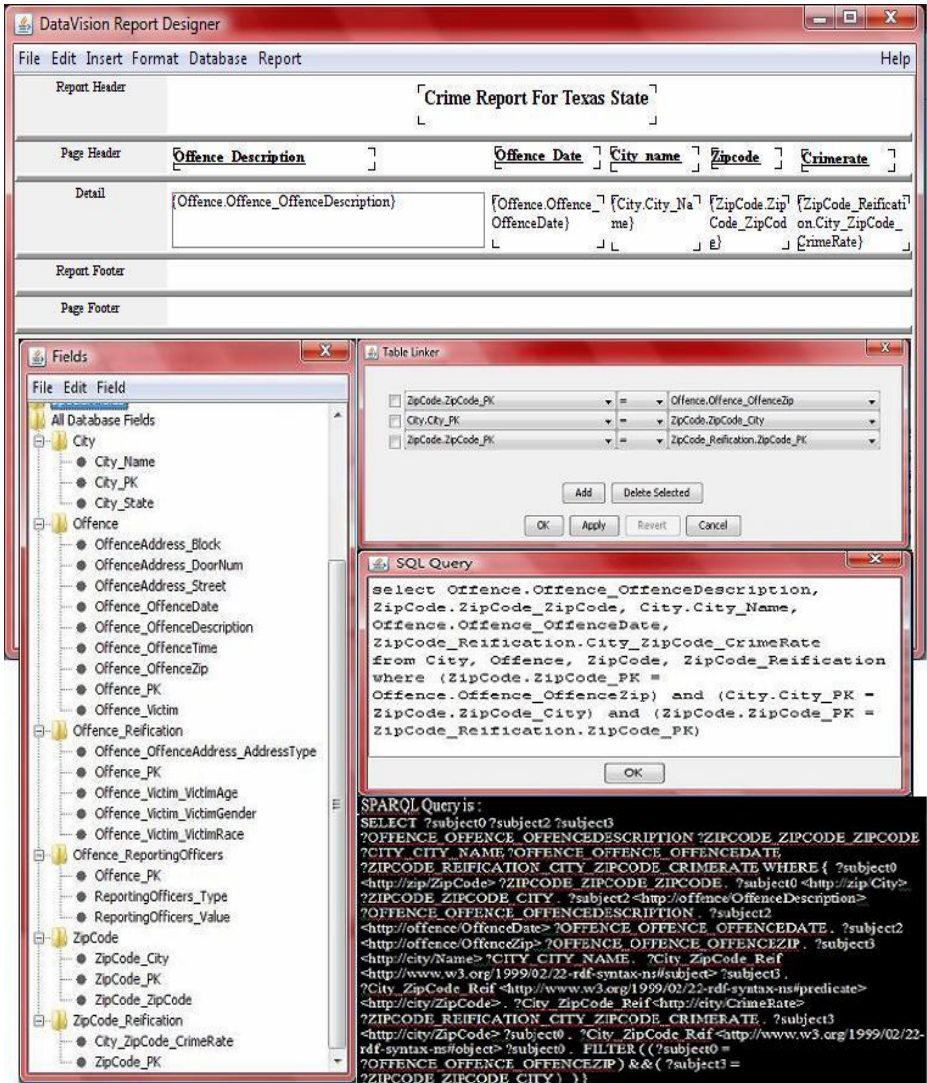


Fig. 5. DataVision Report Designer, Relational Schema, and Query Processing

juncture, the Statement, the Prepared Statement, and the ResultSet JDBC Interfaces are invoked, which in turn trigger the SQL-to-SPARQL translation algorithm and return the obtained results to DataVision in the expected tabular format.

An excerpt from the output returned to DataVision by the SQL-to-SPARQL translation algorithm for the SQL statement in Figure 5 is shown in Figure 6. Selected fields from this output were utilized by another independent application to plot the crime details on Google maps as also illustrated in Figure 6.

In order to study the performance impact incurred by reification two versions of 4 queries were executed on simulated crime datasets of various sizes. The second

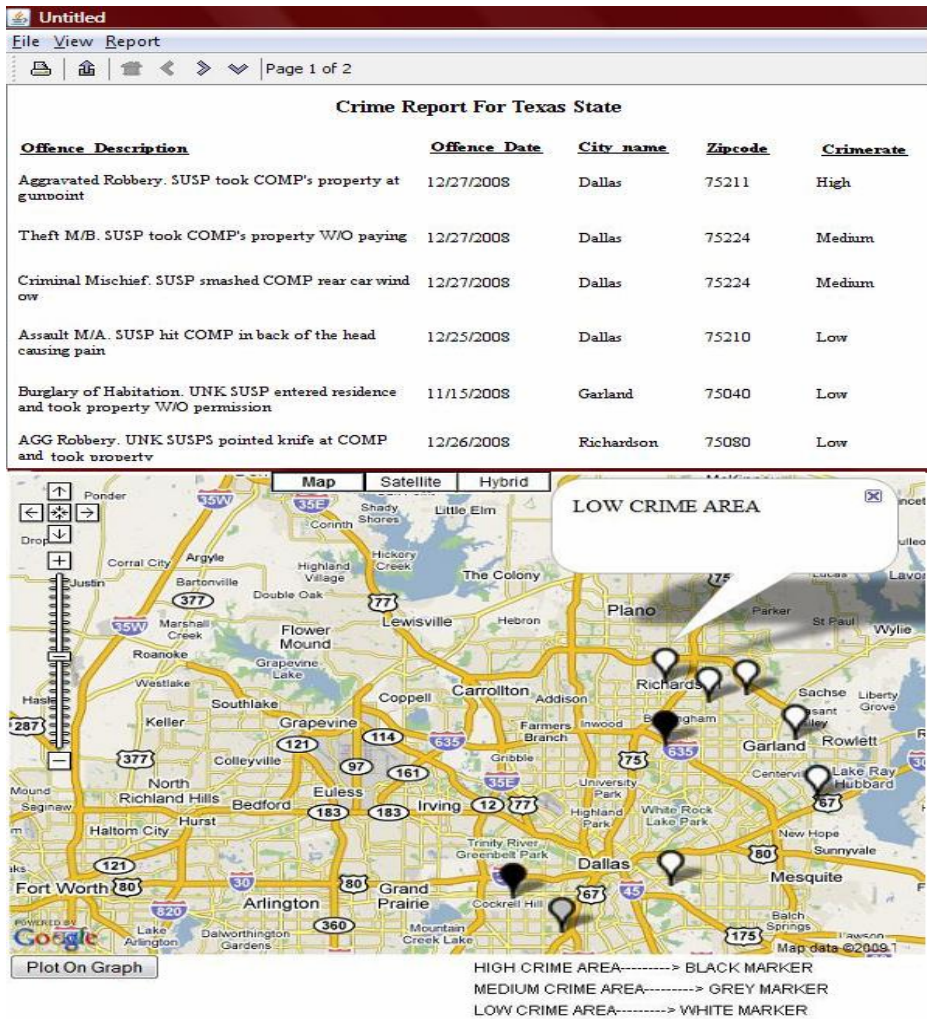


Fig. 6. Excerpt from Datavision's output in report form and Google Maps plot form

version was created by including one or more reification fields to the first version. Figure 7 displays the response times of each of the queries as the sizes of the databases vary. While DataVision has options to specify aggregation and grouping functions, DataVision’s support group has, for reasons that are not applicable to our academic test environment, disabled the GROUP BY facility. For the purposes of our research, we have enabled the functionality.

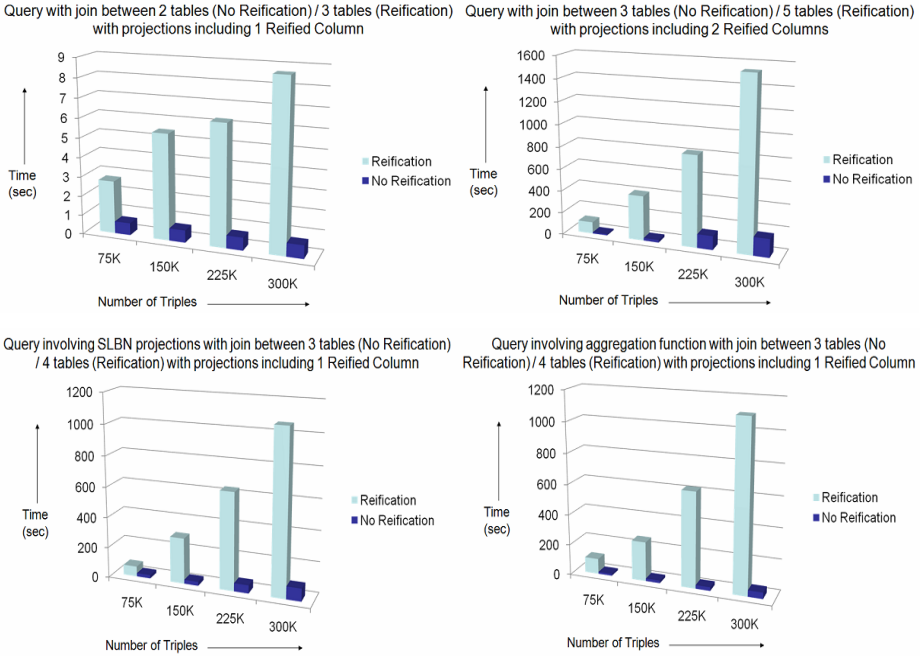


Fig. 7. Response times for the chosen Queries

As was anticipated, reification adds overheads to query processing times as adding a reification quad for a triple results in the addition of a minimum of 4 to 5 extra triples to the data store. However, the time taken for SQL-to-SPARQL conversion is negligible and nearly constant. Thus, R2D does not add overheads to the SPARQL query performance.

SQL queries issued against relational databases created by physically duplicating RDF data may exhibit even better performance since refined performance optimization options have been at the disposal of relational databases for many decades. However, this improved performance comes at the expense of additional disk space due to duplication of data, and additional system resources and human effort required to synchronize the data. On the other hand, for possibly a small price in terms of response time, R2D offers an avenue for users to continue to take advantage of readily available visualization tools without having to “reinvent the wheel”.



## 6 Conclusion

Provenance Information plays a pivotal role in evaluating quality of data and determining trust in the source of data. This paper extends the R2D framework in [3] by including the ability to represent provenance information available in RDF stores, through the process of reification, in a relational format accessible through traditional relational tools. A JDBC interface aimed at accomplishing this goal through a mapping between RDF reification constructs and their equivalent relational counterparts was presented. The modus operandi of the proposed system was described along with in depth discussion on the algorithms comprising the R2D framework. Graphs highlighting response times for map file generation and query processing obtained using databases of various sizes, both with and without reification data, were also included. Future directions for R2D include providing support for the ability to relate an entity key field to multiple `r2d:TableMaps` corresponding to resources belonging to different classes, and improving the normalization process for mixed blank nodes and complex reification nodes.

## References

1. W3C Recommendation, RDF Primer (2004), <http://www.w3.org/TR/rdf-primer/>
2. Hendler, J.: RDF Due Diligence (2006), [http://civicaactions.com/blog/rdf\\_due\\_diligence](http://civicaactions.com/blog/rdf_due_diligence)
3. Ramanujam, S., Gupta, A., Khan, L., Seida, S., Thuraisingham, B.: A Framework for the Relational Transformation of RDF Data. UT Dallas Technical Report UTDCS-40-08 (2008), <http://www.utdallas.edu/~sxr063200/Paper2.pdf>
4. Da Silva Almendra, V., Schwabe, D.: Trust Policies for Semantic Web Repositories. In: Second Semantic Web Policy Workshop, pp. 17–31 (2006)
5. Buneman, P., Chapman, A., Cheney, J.: Provenance Management in Curated Databases. In: Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data, pp. 539–550 (2006)
6. Powers, S.: Practical RDF. O'Reilly Media, Sebastopol (2003)
7. Teswanich, W., Chittayasothorn, S.: A Transformation of RDF Documents and Schemas to Relational Databases. In: IEEE Pacific Rim Conference on Communications, Computers, and Signal Processing, pp. 38–41 (2007)
8. Bizer, C., Cyganiak, R., Garbers, J., Maresch, O., Becker, C.: The D2RQ Platform, <http://www4.wiwiwiss.fu-berlin.de/bizer/d2rq/>
9. Han, L., Finin, T., Parr, C., Sachs, J., Joshi, A.: RDF123: From Spreadsheets to RDF. In: Sheth, A., et al. (eds.) ISWC 2008. LNCS, vol. 5318, pp. 451–466. Springer, Heidelberg (2008)
10. de Laborda, C.P., Conrad, S.: Bringing Relational Data into the Semantic Web using SPARQL and Relational OWL. In: 22nd International Conference on Data Engineering Workshops, p. 55 (2006)
11. Melnik, S.: Storing RDF in a Relational Database, <http://infolab.stanford.edu/~melnik/rdf/db.html>

12. Chebotko, A., Lu, S., Jamil, H.M., Fotouhi, F.: Semantics Preserving SPARQL-to-SQL Query Translation for Optional Graph Patterns. Technical Report TR-DB-052006-CLJF. Wayne State University (2006)
13. Chen, H., Wu, Z., Wang, H., Mao, Y.: RDF/RDFS-based Relational Database Integration. In: 22nd International Conference on Data Engineering, pp. 94–104 (2006)
14. Chong, E.I., Das, S., Eadon, G., Srinivasan, J.: An Efficient SQL -based RDF Querying Scheme. In: 31st International Conference on Very Large Databases, pp. 1216–1227 (2005)

# Employing Key Indicators to Provide a Dynamic Risk Picture with a Notion of Confidence

Atle Refsdal and Ketil Stølen

SINTEF ICT, Norway

Atle.Refsdal@sintef.no, Ketil.Stolen@sintef.no

**Abstract.** A security risk analysis will only serve its purpose if we can trust that the risk levels obtained from the analysis are correct. However, obtaining correct risk levels requires that we find correct likelihood and consequence values for the unwanted incidents identified during the analysis. This is often very hard. Moreover, the values may soon be outdated as the system under consideration or its environment changes. It is therefore desirable to be able to base estimates of risk levels on measurable indicators that are dynamically updated. In this paper we present an approach for exploiting measurable indicators in order to obtain a risk picture that is continuously or periodically updated. We also suggest dynamic notions of confidence aiming to capture to what extent we may trust the current risk picture.

## 1 Introduction

In order for a security risk analysis of a computer system to serve its purpose, we need to trust that the risk levels obtained for the identified risks are (at least roughly) correct. This requires finding good answers to the following questions: 1) How likely is the unwanted incident in question to occur? 2) What is the consequence if this incident occurs? 3) How do the consequence value and the likelihood value combine into a single risk value? Unfortunately, in most cases the answers obtained from a risk analysis will provide a snapshot reflecting a single point in time. Hence, the risk values may soon be outdated as the system or its environment change.

Moreover, finding correct likelihood values is often very hard. This is typically the case if we are analyzing a new system where historical data do not exist, or if the incident in question cannot easily be observed directly, such as an eavesdropper reading a sensitive e-mail. Finding correct consequence values may also be difficult, particularly in cases where the asset we seek to protect is not easily measured in terms of money, such as confidentiality of sensitive data.

All this means that we need to seek ways of obtaining good estimates of likelihood and consequence values. One way of doing this is to base the assessments on measurable indicators that are seen as relevant for the unwanted incident in question, even though its likelihood or consequence value cannot be directly inferred from any of these indicators. For example, if we want to estimate the likelihood that an intruder accesses sensitive data by logging on to a system with the username and password of a legitimate user, it may be useful to know how many passwords have not been changed during the last

three months and how many of the users do not comply with the company's password strength policy. If likelihood, consequence and risk levels are defined as functions of indicators, we also ensure that risk levels can be updated as soon as the indicators are updated, rather than representing a snapshot at a given point in time.

Having assigned likelihood values to the relevant threat scenarios and unwanted incidents, we are also interested in estimating the level of confidence we may have that the risk levels obtained are in fact correct, and to uncover weaknesses of the analysis. One way of achieving this is to check whether the risk picture is consistent with respect to likelihood values. This can be done by assigning likelihood not only to the unwanted incident that harms an asset, but also to the potential threat scenarios that may lead to this incident. The likelihood of the unwanted incident can then be compared to the likelihood of the threat scenarios.

This paper presents an approach for providing a dynamic risk picture and for assessing to what degree we can be confident that the risk levels obtained are correct. A basic assumption of the approach is that an infrastructure is available for defining and monitoring the measurable indicators required. Providing such an infrastructure is an important goal for the project MASTER (see <http://www.master-fp7.eu/>), which addresses the challenge of managing assurance, security and trust for service-oriented systems. Although the work presented has been carried out within the context of the MASTER project, the approach we present is general in the sense that we just assume the availability of a palette of monitored indicators; the infrastructure required to obtain them is not considered.

The rest of this paper is structured as follows: In Section 2 we present the conceptual model on which the approach is based. A high-level description of the approach, as well as the underlying assumptions, is given Section 3, which ends with presenting three steps that need to be performed in order to carry out the dynamic risk monitoring. Based on an example case, these three steps are further explained in Sections 4, 5, 6. We then explain how the internal consistency of the risk picture can be checked in Section 7, and discuss measures of confidence in the analysis in Section 8. Some related work is presented in Section 9, before we conclude in Section 10.

## 2 Conceptual Model

Figure 1 shows the conceptual model for risk and closely related concepts on which our approach is based. The model is shown as a UML class diagram with explanatory text on the associations. A risk involves an unwanted incident, such as sensitive patient data being disclosed to outsiders. The unwanted incident may occur with a certain likelihood. When it occurs, an asset will be damaged (and its value reduced) – this is the consequence of the risk. An asset is something of value that we seek to protect. Assets can be anything from physical objects such as computers to abstract entities such as confidentiality of information or the reputation of a stakeholder. If the asset we are concerned with is the reputation of the hospital, and the identified incident is sensitive patient data being disclosed to outsiders, then the consequence related to this incident could be a certain reduction of (or damage to) the hospital's reputation. In the diagram, we have assigned consequence directly to the risk, rather than to the unwanted incident.

This has been done in order to emphasize that the consequence of an incident, unlike its likelihood, is not a property of the incident per se, as the consequence also depends on the particular asset in question and its measure.

In order to obtain a clear risk picture and be able to choose and prioritize between treatments, we need to assign a risk value to each risk that has been identified. This is done by applying the risk function, which takes the consequence and the likelihood of the unwanted incident of the risk as input. Hence, consequence and likelihood need to be measured according to some suitable scale. Typically, likelihood is measured in terms of frequency or probability. Consequence may be measured by for example monetary value or the number of data items affected by the incident, dependent on the nature of the asset in question. The risk function is defined by the risk analysis team and depends on the scales chosen for measuring consequence and likelihood. If we are measuring likelihood in terms of frequency and consequence in terms of monetary value, we may use multiplication to obtain a risk value. For example, from a consequence of 10000 euros and a likelihood of 3 times per year we get a risk value of 30000 euros per year. If qualitative scales are used for measuring consequence and likelihood, then the risk function defines how the possible consequence and likelihood values combine into a risk value. For example, a consequence value “catastrophic” and a likelihood value “seldom” may combine into a risk value “high”.

As discussed in Section II obtaining correct consequence and likelihood values is a major challenge. Therefore it may be highly useful to be able to identify relevant and measurable key indicators on which estimates can be based. Notice that the multiplicity symbol \* on each end of the associations between the KeyIndicator class and the Likelihood class in Figure I means that there is a many-to-many relationship between key indicators and likelihoods; one likelihood may be obtained from several key indicators, and one key indicator may be used to obtain the likelihood of several unwanted incidents. The same holds for the relation between key indicators and consequences.

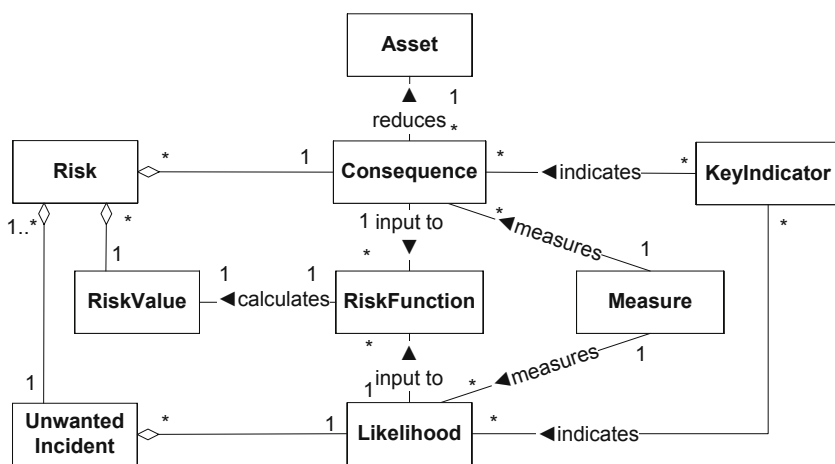


Fig. 1. Conceptual model for risk and closely related concepts

### 3 The Approach

Figure 2 outlines our vision for a dynamic risk monitor defined on the top of some monitoring infrastructure. In this paper we just assume the availability of a monitoring infrastructure offering a palette of continuously monitored key indicators that we may select from. The key indicators are quantitative measures that are considered relevant for finding the likelihood or consequence of an unwanted incident. In many cases, indicator values will be calculated automatically by the system. For example, the system can recognize a certain kind of event and count the number of occurrences of such events. In other cases, the indicator values will be obtained manually. For example, the number of errors detected in a periodic review of a sample of the records stored in a database may be input into the system to provide an indicator value. Our envisaged dynamic Risk Monitor consists of three modules as indicated in Figure 2, and the rest of this paper is devoted to establishing the data required for it to function, given the assumed availability of a palette of key indicators.

The “Dynamic Risk Picture” module allows the user to monitor the likelihood, consequence, and risk values, thereby providing a more high-level view than the “Key Indicators” infrastructure. Values may be presented in graphical diagrams that show how threat scenarios lead up to unwanted incidents; likelihood values may be assigned to threat scenarios as well as unwanted incidents. The values are obtained from functions for calculating likelihood and consequence values from sets of key indicators, as well as for calculating risk values from likelihood and consequence values. These functions are defined during the risk analysis of the system, as the relevant risks will depend on the system in question.

The “Risk Consistency” module checks whether the risk picture is consistent at a given point in time. This can be done by comparing likelihoods for threat scenarios assumed to lead up to an unwanted incident with the likelihood of the actual incident. For example, if a certain threat scenario is assigned probability “twice a year”, and is assumed to lead to a certain unwanted incident with probability 0.5, then the likelihood assigned to the unwanted incident should be at least “once a year”. But it can also be higher, if there are other threat scenarios leading up to the same incident. In Section 7 we present calculation rules taken from [BDS08] that can be utilized in order to check whether the likelihood values assigned to a diagram are consistent.

Finally, the “Confidence” module offers a quantitative measure of confidence in the current risk picture, thereby providing an aggregated view from which the correctness

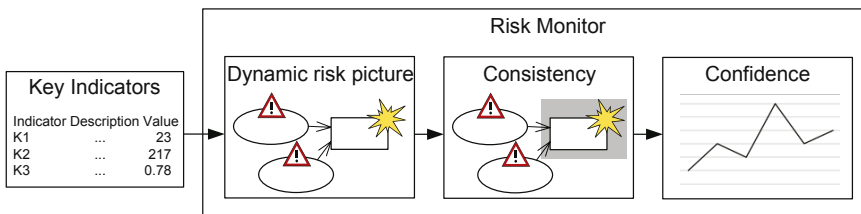


Fig. 2. Risk Monitor modules

of the analysis can be assessed. The aim is to estimate to what extent we may trust that the risk levels are correct based on the degree of inconsistency detected in the risk picture.

The programming of the dynamic risk monitor may of course be work consuming, but not very challenging from a research point of view. The real issue of research is how to come up with the data to display, and this is what we concentrate on in the rest of the paper. We propose an approach of three steps that need to be carried out before monitoring can start, which we will describe in further detail in the next sections:

1. *Perform an initial risk analysis of the system.* This step serves a number of purposes. It provides information about what are the relevant risks and a rough analysis of the risk levels, so that a decision can be made of which risks need to be monitored. Furthermore, it provides information about how threats exploit vulnerabilities to initiate threat scenarios leading to unwanted incidents, which is essential for the later steps.
2. *Identify relevant key indicators for the risks to be monitored.* This is done based on the understanding obtained through the initial analysis in the previous step. Indicators may be related not only to an unwanted incident that is directly associated with a risk, but also to vulnerabilities and threat scenarios leading up to this incident.
3. *Find functions for likelihood, consequence, and risk values.* Likelihood and consequence values are calculated from sets of key indicators, while risk values are calculated from the likelihood and consequence value for the unwanted incident in question.

## 4 Performing the Initial Risk Analysis of the System

We consider a hospital concerned about protecting the integrity of patient records. All details have been made up. Hence, the unwanted incidents, threat scenarios, risk levels and other aspects of the analysis presented here do not reflect any real case.

The patient record database can only be accessed from terminals in the hospital's office area. Access to the terminals is protected by user names and passwords. A password strength policy has been issued informing employees of requirements with respect to passwords length, use of numerical characters and so on. In addition, users are expected to change their password every third month. Users do not in general have access to all the patient records. For example, a doctor has only access to the records of her/his own patients. Before leaving a terminal, users are supposed to log off. Users are logged off automatically if a terminal has been inactive for a certain time (the delay logoff interval).

The doors into the office area are normally kept closed and locked at all times, and are fitted with keycard locks. In order to open a door, a keycard has to be inserted into the lock. The door will then open up and remain open for a certain interval in order to allow entry. If the keycard lock of a door is defective the door will unlock. Keycard locks are fitted with a failure detection system that generates a signal if the lock is defect.

The asset that we seek to protect for the purpose of this example is integrity of patient data. There is, of course, many ways in which we can imagine that this asset may be

**Table 1.** Consequence scale (left) and likelihood scale (right)

Consequence Description		Likelihood	Description
Catastrophic	> 400 records affected	Very often	> 100 times per year
Major	101-400 records affected	Often	21 – 100 times per year
Moderate	21-100 records affected	Sometimes	6 – 20 times per year
Minor	3-20 records affected	Seldom	3 – 5 times per year
Insignificant	0-2 records affected	Very seldom	≤ 2 times per year

**Table 2.** Risk evaluation matrix

		Consequence				
		<i>Insignificant</i>	<i>Minor</i>	<i>Moderate</i>	<i>Major</i>	<i>Catastrophic</i>
Frequency	<i>Very seldom</i>					
	<i>Seldom</i>					
	<i>Sometimes</i>					
	<i>Often</i>					
	<i>Very often</i>					

harm. In order to limit the scope, the analysis will be restricted to external threats, and we consider only cases where such data are accessed by intruders into the office area that are not part of the hospital staff. To conduct the initial risk modeling and analysis of the system we employ CORAS. However, other approaches may also be used.

CORAS [dBHL<sup>+</sup>07] provides a method, a language, and a tool for asset-oriented risk analysis. The CORAS method consists of seven steps. For the purpose of this paper, we focus on only a few of these. In order to assign likelihood and consequence values to unwanted incidents, we need to establish some suitable scales for this that are useful for making assessments. Table 1 shows the scales that will be used to measure consequence for the “Integrity of patient records” asset in this paper, as well as the scale that will be used for measuring likelihood. Note that by choosing to measure consequence for integrity of patient data only in terms of the number of records affected, we do not distinguish between different levels of importance for different records. If necessary, we could have identified separate assets for records based on their importance.

After deciding upon the suitable scales for consequence and likelihood, the analysts establish the risk evaluation criteria that states which level of risk the client accepts for each asset. The result is typically recorded as a risk matrix that shows which combinations of consequence and likelihood values that are acceptable and which are not. Table 2 shows such a matrix. A gray box means that the risk level is so high that the risk needs to be further evaluated for treatment.

After establishing the risk evaluation criteria, the next steps concern identifying potential unwanted incidents and the scenarios leading up to these incidents. The result is documented in *threat diagrams*. Threat diagrams show how threats exploit vulnerabilities to initiate threat scenarios and unwanted incidents, and what assets are harmed if the unwanted incident occurs. A threat scenario is a scenario that may lead to an unwanted incident or to another threat scenario. Figure 3 shows the symbols used to denote threats





Fig. 3. Basic building blocks of CORAS threat diagrams

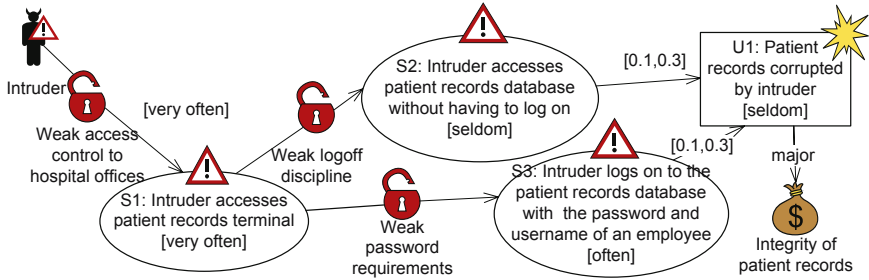


Fig. 4. Threat diagram with likelihood and consequence estimates

(of three different kinds), vulnerabilities, threat scenarios, unwanted incidents and assets. Except from vulnerabilities, these elements are referred to as vertices. Figure 4 shows a threat diagram for the hospital example. Note that this is not intended to represent a complete analysis. In addition to the symbols shown in Figure 3, a threat diagram contains relations represented by arrows between the vertices, possibly via one or more vulnerabilities. These vulnerabilities are then considered to be a part of the relation. Hence, a threat diagram consists of a set of vertices and a set of relations between the vertices. For a formal definition of the language, see [BDS08]. There are three kinds of relations: “initiate”, “leads-to” and “impact”. An “initiate” relation goes from a threat to a threat scenario or an unwanted incident, and shows that the threat initiates the threat scenario or unwanted incident. Possibly, the threat achieves this by exploiting one or more vulnerabilities, which are then shown on the arrow from the threat to the threat scenario or unwanted incident.

The “initiate” relation in the left-hand part of Figure 4 shows that the threat “Intruder” exploits the vulnerability “Weak access control to hospital offices” to initiate the threat scenario “S1: Intruder accesses patient records terminal”. From threat scenario S1 there is a “leads-to” relation showing that this scenario may lead to the threat scenario S2 via the vulnerability “Weak logoff discipline”. This means that the intruder accesses a terminal where the previous user has not logged off before leaving the terminal. The “leads-to” relation from S2 show that this scenario may lead to the unwanted incident U1, which impacts the asset “Integrity of patient records”. From threat scenario S1 there is also another “leads-to” relation to S3 showing that the intruder may achieve the same unwanted incidents by logging on to a terminal with the user name and password of an employee.

<sup>1</sup> We use S1, S2, S3 as shorthand names for threat scenarios, and U1 for the unwanted incident.

Having identified the unwanted incidents, threats, vulnerabilities, and threat scenarios, as well as the “initiate”, “leads-to” and “impact” relations between them, the next step is to assign likelihood and consequence values. This has also been done in Figure 4. Likelihood values are inserted in brackets on the threat scenarios and unwanted incidents, while consequence values are inserted on the “impact” relations from unwanted incidents to assets. Probability intervals have also been assigned to the “leads-to” relations from threat scenarios to unwanted incidents.

### 5 Identifying Relevant Key Indicators

In order to calculate and monitor risk values based on key indicators, we first need to identify the indicators that are of relevance for the risks in question. As seen in Section 4, CORAS threat diagrams illustrate graphically how unwanted incidents result from threats exploiting vulnerabilities to initiate threat scenarios. These diagrams can be exploited in a structured brain storming in order to identify relevant key indicators. The analysis leader can direct the attention of the analysis team to the different elements of the diagram one at a time, each time asking for suggestions for suitable indicators and noting these down at the relevant place in the diagram. Thereby, the team is encouraged to think about not only indicators directly associated with the unwanted incident, but also indicators that are more closely related to vulnerabilities and scenarios leading up to the incident. Figure 5 shows a possible result of applying this process on the diagram in Figure 4. Indicators have been chosen in order to illustrate different aspects of the approach, and do not represent a real (or exhaustive) analysis. Key indicators are shown as boxes with a dial in the upper right-hand corner, and are attached to vulnerabilities, threat scenarios, unwanted incidents and “impacts” relations. The indicators K1 “Time interval for opening of doors” and K2 “Total time that keycard locks have been defect during the last 3 months

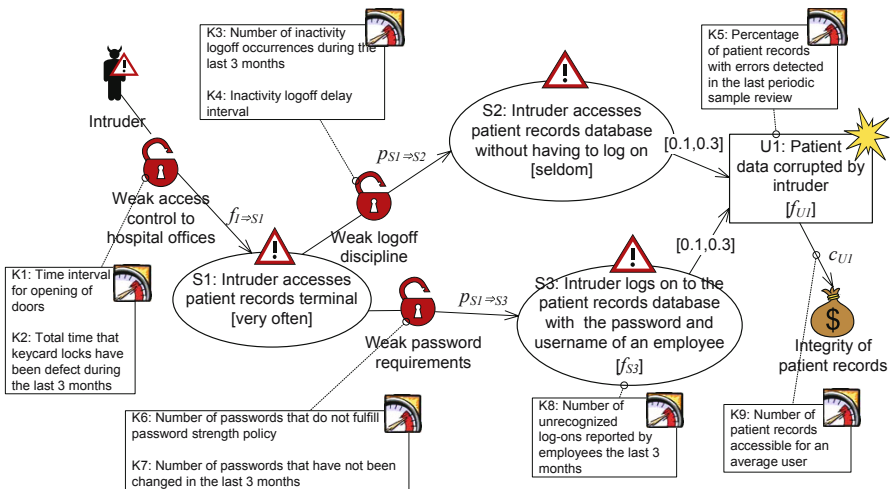


Fig. 5. Threat diagram with indicators attached

been defect during the last 3 months” have been attached to the vulnerability “Weak access control to hospital offices”. The reasoning is that it will be easier for an intruder to access the office area if doors remain open for a relatively long time after someone has entered or left, or if a keycard lock is defect so that the door is unlocked. For the vulnerability “Weak logoff discipline”, the indicators  $K3$  and  $K4$  have been identified as relevant, measuring how often it occurs that users are automatically logged off due to inactivity, and the length of the logoff delay interval, respectively. For assessing the likelihood of the unwanted incident “Patient data corrupted by intruder”, we assume that periodic reviews are held of random samples of the patient records. The doctors of the patients in question are asked to go through the records to check that the data are correct. For example, recorded treatments should match the patient’s disease. The indicator  $K5$  measures the percentage of records with errors reported by doctors in the sample review. The indicators  $K6$  “Number of passwords that do not fulfill password strength policy” and  $K7$  “Number of passwords that have not been changed in the last 3 months” have been identified for the vulnerability “Weak password requirements”. For threat scenario  $S3$ , the indicator  $K8$  has been introduced as an aid in assessing the likelihood. The idea is that a message with the date and time of the previous log-on pops up each time a user logs on, and the user is asked whether she or he can confirm that this is correct. The indicator  $K8$  measures the number of occurrences where users give a negative answer during the last three months. Finally, the indicator  $K9$  measures the number of patient records that are accessible for an average user. This indicator is attached to the “impacts” relation from the unwanted incident “Patient data corrupted by intruder” to the asset “Integrity of patient records”, showing that this indicator can be used as an aid in assessing the consequence of this incident.

Note that likelihood and consequence values have been replaced by function names for the vertices and relations for which indicators have been identified. These functions are explained in the next section.

## 6 Finding Functions for Likelihood, Consequence, and Risk Values

After identifying relevant key indicators for the risks to be monitored, the next step is to define functions for calculating likelihood and consequence values from the indicators. This is done for all the vertices and relations for which relevant indicators have been identified. For example, for the “initiate” relation from the threat “Intruder” to  $S1$  we need to define a function that calculates the frequency with which an intruder initiates this threat scenario from the indicators  $K1$  and  $K2$ .

We use function names with subscripts to show which function we are dealing with, according to the following convention: The first letter denotes the type of the output of the function;  $f$  for frequency,  $p$  for probability, and  $c$  for consequence. The subscript denotes which frequency/probability/consequence we are talking about, and we use  $\rightarrow$  in the subscript when referring to a relation between two vertices. For example,  $f_{I \rightarrow S1}(K1, K2)$  denotes the function for calculating the frequency with which the intruder initiates  $S1$  ( $I$  is shorthand for “Intruder”) from  $K1$  and  $K2$ ,  $p_{S1 \rightarrow S3}(K6, K7)$  denotes the function for calculating the conditional probability that  $S1$  leads to  $S3$  from  $K6$  and  $K7$ , and  $f_{S3}(K8)$  denotes function for calculating the frequency of  $S3$  from  $K8$ .

**Table 3.** Key indicators with values at the time of the initial analysis

Name	Description	Domain	Unit	Value
<i>K1</i>	Time interval for opening of doors.	{2, ..., 30}	seconds	16
<i>K2</i>	Total time that keycard locks have been defect during the last 3 months.	{0, ..., 2160}	hours	24
<i>K3</i>	Number of inactivity logoff occurrences during the last 3 months.	{0, ...}	-	21
<i>K4</i>	Inactivity logoff delay interval.	{5, ..., 30}	minutes	8
<i>K5</i>	Percentage of patient records with errors detected in the last periodic sample review.	[0, 1]	-	0.01
<i>K6</i>	Number of passwords that do not fulfill password strength policy.	{0, ..., 200}	-	40
<i>K7</i>	Number of passwords that have not been changed in the last 3 months.	{0, ..., 200}	-	35
<i>K8</i>	Number of unrecognized log-ons reported by employees the last 3 months.	{0, ..., 200}	-	6
<i>K9</i>	Number of patient records accessible for an average user.	{0, ..., 2500}	-	224

When defining the functions for the vertices and relations where indicators have been attached, we may get some guidance from the values of the indicators at the time when the initial analysis was performed, as the functions should give likelihood (and consequence) values in the intervals obtained in the initial analysis when applied on these values. Table 3 shows the indicator values that we assume apply at the time of the initial analysis in our example. Exactly how to define the functions depends on each particular case, and must be based on the expertise and judgment of the analysis team, as well as existing statistical data if available. We now describe how it might be done for the example. Note that the definitions below have been made up in order to illustrate the approach.

We start with the function  $f_{I \rightarrow S1}(K1, K2)$ . From Figure 4 we see that an initial estimate has been made for the current value of this function. The analysts therefore note that the function should, when applied to the above values for *K1* and *K2*, give a frequency value “very often”, which corresponds to more than 100 times per year.

Based on existing data about physical access control, their own experience in the field, and knowledge about the hospital in question, the analysts expect that if a door is unlocked (due to a defect keycard lock), there will be on average one intruder every day, or 365 intruders per year. As three months equals 2160 hours, and *K2* gives the number of hours a keycard lock is defect during a three month period, the contribution to  $f_{I \rightarrow S1}(K1, K2)$  due to defect keycard locks is  $\frac{K2}{2160} \times 365$ . Furthermore, the analysts expect that as long as the keycard locks function properly, the number of intruders per year will be proportional to the opening interval of doors. In the worst case, where doors remain open for 30 seconds after being opened, the analysts consider that it will be almost as easy to gain access as when a keycard lock is broken, as there is a lot of traffic in and out of the hospital; they therefore expects 300 intruders per year in this case. In the best case, where doors only remain open for 2 seconds after being opened,

it will be much harder to gain access by following after a hospital employee. In this case the analysts expect 20 intruders per year. This gives the contribution  $(1 - \frac{K2}{2160}) \times 10 \times K1$  for the time periods when no keycard locks are defect. All in all, the above considerations give the following function:

$$f_{I \rightarrow S1}(K1, K2) = \frac{K2}{2160} \times 365 + (1 - \frac{K2}{2160}) \times 10 \times K1 \quad (1)$$

Applying this function on the arguments  $K1 = 16$  and  $K2 = 24$  gives a value of 162 per year, which is indeed in accordance with the initial estimate.

For defining the function  $p_{S1 \rightarrow S2}(K3, K4)$ , the analysts assume that the value will be at least 0.01, and at most 0.95. Within these limits, the probability is expected to be proportional with the product of  $K3$  and  $K4$  and be given by  $\frac{K3 \times K4}{x}$  for some suitable  $x$ , as the product of  $K3$  and  $K4$  says something about the length of time during the observation period there will be terminals where the user have forgotten to log off. Clearly, the value of  $x$  has to be set so that a suitable probability is obtained. The analysts decide to estimate this number from the case where  $K3 = 300$  and  $K4 = 10$ . In this case, the value of  $p_{S1 \rightarrow S2}(K3, K4)$  is expected to be 0.5. Hence, the value of  $x$  is given by  $\frac{300 \times 10}{x} = 0.5$ , which gives  $x = 6000$ . The analysts confirm that this is a suitable value for  $x$  by inserting alternative values for  $K3$  and  $K4$ , in each case verifying that the resulting probability is within their expectations. These considerations give the following definition of  $p_{S1 \rightarrow S2}(K3, K4)$ :

$$p_{S1 \rightarrow S2}(K3, K4) = \begin{cases} 0.01 & \text{if } \frac{K3 \times K4}{6000} \leq 0.01 \\ 0.95 & \text{if } \frac{K3 \times K4}{6000} \geq 0.95 \\ \frac{K3 \times K4}{6000} & \text{otherwise} \end{cases} \quad (2)$$

For defining the function  $p_{S1 \rightarrow S3}(K6, K7)$  for calculating the conditional probability that  $S1$  leads to  $S3$  from  $K6$  and  $K7$ , the analysts assume that the value will be at least 0.01, which is the case where all users change their password every third month and follow the password strength policy (i.e. when  $K6 = K7 = 0$ ), and at most 0.7, which is the case when none of the users do this (i.e. when  $K6 = K7 = 200$ ). The analysts consider an old password to be just as bad as a weak password (and a password that is both old and weak to be twice as bad), and expects that the probability will depend linearly on the sum of  $K6$  and  $K7$  between these limits. This gives the following definition:

$$p_{S1 \rightarrow S3}(K6, K7) = \frac{0.69 \times (K6 + K7)}{400} + 0,01 \quad (3)$$

Note that, according to the above definitions and the possible values of the indicators, the probability that  $S1$  leads to  $S2$  and the probability that  $S1$  leads to  $S3$  may sum up to more than 1. The reason for this is that we assume that an intruder may initiate both  $S2$  and  $S3$ , so that they are not mutually exclusive. For example, when gaining access to a terminal where the previous user has forgotten to log off, the intruder may decide to try to log on as a different user in order to gain access to patient records that are not accessible for the previous user. In general, CORAS diagrams do not require that the sum of probabilities for the outgoing relations from a vertex should add up to 1 or

less, as one scenario may lead to a number of different scenarios or unwanted incidents simultaneously.

For calculating the frequency of  $S3$  from  $K8$ , the analysis team assumes that the number of unrecognized log-ons reported by employees (and measured by  $K8$ ) reflects the actual situation reasonably well. Therefore, they decide to simply use the value of  $K8$  multiplied with 4 to obtain the number of occurrences per year, rather than per 3 months.

$$f_{S3}(K8) = K8 \times 4 \quad (4)$$

Inserting the current value of  $K8 = 6$ , this gives a frequency of 24 times per year, which is in accordance with the initial estimate “Often”, i.e. from 21 to 100 times per 10 years.

For calculating the frequency of  $U1$  from  $K5$ , the analysts decide to set a minimum value of 2 per ten years, independently of  $K5$ . This is done in order to avoid a situation where the risk value associated with  $U1$  becomes 0 in the cases where the frequency is so low that it might not be captured by the periodic sample review. Above this minimum value, the frequency is assumed to be proportional with  $K5$ . In order to obtain a frequency from the probability  $K5$ , the analysts reason as follows: There are 2500 patient records in all, so the number of records with errors are  $2500 \times K5$ . The records have an average age of 5 years, and errors are assumed to have been introduced during the last 5 years, which means that the frequency of error introduction is  $\frac{2500 \times K5}{5}$  per year. These considerations give the following function:

$$f_{U1}(K5) = \begin{cases} 2 & \text{if } K5 \times 500 < 2 \\ K5 \times 500 & \text{otherwise} \end{cases} \quad (5)$$

Inserting the current value of  $K5 = 0.01$ , this gives a frequency of 5 times per year, which is in accordance with the initial estimate “Seldom”, i.e. from 3 to 5 times per year.

The last place in the diagram where an indicator has been identified is the “Impacts” relation from  $U1$  to the asset “Integrity of patient data”. It therefore remains to define a function that calculates the consequence of  $U1$  from the relevant indicator  $K9$ . For an incident where data is corrupted, the consequence may clearly depend on the nature of the corruption. The intruder may, for example, add false information or delete some or all fields of one or more records. However, it was decided in the initial analysis that consequence should be measured simply in the number of records affected. The analysts decide to assume that the intruder, when corrupting patient data, manages to corrupt all the records available for the user in question. As  $K9$  measures the number of patient records available for an average user, they therefore decide to define the function simply as follows:

$$c_{U1}(K9) = K9 \quad (6)$$

## 7 Evaluating Consistency

Obtaining a correct threat diagram with suitable indicators and functions for calculating likelihood values from indicators will clearly be quite challenging. As illustrated by the previous section, the subjective judgments made by experts and analysts will typically

play a major role. It is therefore important to have ways of discovering weaknesses and aspects that need to be reconsidered. Being able to automatically check whether the values obtained are consistent is therefore important. By consistent we mean that the likelihood value assigned to a vertex should match the value that can be obtained from the likelihood values of the preceding vertices and incoming “leads-to” relations. For example, the frequency of  $S3$  should match the frequency we obtain from the frequency of  $S1$  and the probability assigned to the “leads-to” relation from  $S1$  and  $S3$ . The CORAS calculus introduced in [BDS08] provides rules that can be employed to calculate the likelihood of a vertex indirectly in this way. This allows us to check the consistency of likelihood values for the vertices where we have one value calculated directly from the indicators attached to the vertex and another value calculated indirectly from preceding vertices and “leads-to” relations. In our example this is the case for  $S3$  and  $U1$ .

Before showing how the consistency for these two vertices can be analyzed, we now explain some of the most important rules of the CORAS calculus. For a more comprehensive presentation, see [BDS08]. In the following we use  $t$  to denote a threat, while  $v, v_1, v_2$  denote vertices that may be threat scenarios or unwanted incidents. We use  $t \xrightarrow{f} v$  to denote that  $t$  initiates  $v$  with frequency  $f$ , while  $v_1 \xrightarrow{p} v_2$  denotes that  $v_1$  leads to  $v_2$  with conditional probability  $p$ .  $v(f)$  denotes that vertex  $v$  occurs with frequency  $f$ .

The “initiate” rule captures the semantics of the “initiate” relation. The frequency of the occurrences of vertex  $v$  due to threat  $t$  is equal to the frequency with which  $t$  initiates  $v$ . This is captured by the following rule, where  $t \sqcap v$  can be understood as “the subset of the scenarios/incidents  $v$  initiated by threat  $t$ ”.

**Rule 1 (Initiate).** For threat  $t$  and vertice  $v$  related by “initiate”, we have:

$$\frac{t \xrightarrow{f} v}{(t \sqcap v)(f)}$$

The “leads-to” rule captures the conditional probability semantics embedded in the “leads-to” relation. The likelihood of the occurrences of  $v_2$  that are due to  $v_1$  is equal to the frequency of  $v_1$  multiplied with the conditional probability that  $v_1$  will lead to  $v_2$  given that  $v_1$  occurs. This is captured by the following rule, where  $v_1 \sqcap v_2$  can be understood as “the subset of the scenarios/incidents  $v_2$  that result from  $v_1$ ”.

**Rule 2 (Leads-to).** For the vertices  $v_1$  and  $v_2$  related by “leads-to”, we have:

$$\frac{v_1(f) \quad v_1 \xrightarrow{p} v_2}{(v_1 \sqcap v_2)(f \times p)}$$

If two vertices are mutually exclusive the likelihood of their union is equal to the sum of their likelihoods. This is captured by the following rule, where  $v_1 \sqcup v_2$  denotes all instances of the scenarios/incidents  $v_1$  and  $v_2$ .

**Rule 3 (Mutually exclusive vertices).** If the vertices  $v_1$  and  $v_2$  are mutually exclusive, we have:

$$\frac{v_1(f_1) \quad v_2(f_2)}{(v_1 \sqcup v_2)(f_1 + f_2)}$$

Finally, if two vertices are statistically independent, the likelihood of their union is equal to the sum of their individual likelihoods minus the likelihood of their intersection.

**Rule 4 (Independent vertices).** *If the vertices  $v_1$  and  $v_2$  are statistically independent, we have:*

$$\frac{v_1(f_1) \quad v_2(f_2)}{(v_1 \sqcup v_2)(f_1 + f_2 - f_1 \times f_2)}$$

To illustrate the use of the consistency rules, we now assume that some time has passed after the initial analysis, and that the indicator values have changed to the following values:  $K1 = 17, K2 = 8, K3 = 23, K4 = 8, K5 = 0.03, K6 = 61, K7 = 72, K8 = 5, K9 = 230$ . Note that the initial likelihood and consequence estimates assigned in Figure 4 are outdated at this point, as they applied at the time of the initial analysis.

In order to simplify the presentation, in the following we set  $p_{S2 \rightarrow U1} = p_{S3 \rightarrow U1} = 0.15$ . As no indicators were identified for these probabilities, the value will not change. Furthermore, we assume that the diagram in Figure 4 is meant to be complete, in the sense that there are no other threats or threat scenarios that may initiate or lead to any of the described threat scenarios or unwanted incidents. This means that all instances of  $S1$  are initiated by the threat Intruder (denoted by  $I$ ), and hence that  $S1 = I \sqcap S1$ . Furthermore, it means that all occurrences of  $S2$  and  $S3$  are due to  $S1$  (i.e. that  $S2 = S1 \sqcap S2$  and  $S3 = S1 \sqcap S3$ ), and that all occurrences of  $U1$  are due to  $S2$  or  $S3$ , i.e. that  $U1 = (S2 \sqcap U1) \sqcup (S3 \sqcap U1)$ . This completeness assumption allows us to view the likelihood estimates obtained through use of the above rules as actual values, rather than lower limits.

We first look at the consistency of the frequency of  $S3$ . This value can be obtained either indirectly from the preceding vertices and relations by application of the above rules, or directly from (4). With the new value for  $K8$ , the latter approach gives the frequency 20 per year for  $S3$ . Taking the indirect approach, we start by calculating the frequency with which the intruder initiates  $S1$ . The new values for  $K1$  and  $K2$  gives  $f_{I \rightarrow S1}(K1, K2) = 171$ . We then apply Rule 1 to obtain the frequency 171 per year for  $S1$ . Now we want to apply Rule 2 to calculate the frequency of  $S3$  from the frequency of  $S1$ . First we calculate  $p_{S1 \rightarrow S3}(K6, K7) = 0.24$  from the new indicator values. Rule 2 then gives us the frequency  $171 \times 0.24 = 41$  per year for  $S3$ . Hence, we have a difference of 21 per year for the two estimates of frequency for  $S3$ . For simplicity, in the following calculations we will use  $S3 = 41$  rather than  $S3 = 20$ , but we could also have tried both values in order to see which gives the highest degree of consistency.

Next, we want to check the consistency of frequency estimates for  $U1$ . Again, taking the direct approach is easy; applying (5) on the new value of  $K9$ , we get the frequency 15 per year for  $U1$ . For the indirect approach, we note that both  $S2$  and  $S3$  may lead to  $U1$  according to Figure 4. However, as  $S2$  and  $S3$  are not considered to be mutually exclusive or statistically independent, we cannot obtain an exact value from Rule 3 or Rule 4. We are therefore confined to calculating maximum and minimum values.

Clearly, the minimum frequency of  $U1$  cannot be lower than the highest of the frequencies we obtain from coming to  $U1$  from one of  $S2$  or  $S3$ , i.e. the frequency of either  $S2 \sqcap U1$  or  $S3 \sqcap U1$ . To find the former we first need to calculate the frequency of  $S2$ . Applying Rule 2, we obtain this value by calculating the frequency of  $S1$  with the probability  $p_{S1 \rightarrow S2}(K3, K4)$  that  $S1$  leads to  $S2$ , which with the new indicator values gives



the frequency  $171 \times 0.03 = 5$  times per year for  $S2$ . Applying Rule 2 on the “leads-to” relation from  $S2$  to  $U1$  then gives a frequency of ca 1 per year for  $S2 \sqcap U1$ . Similarly, we use Rule 2 to obtain a frequency of ca 6 per year for  $S3 \sqcap U1$ . Hence, according to these calculations the minimum frequency of  $U1$  is 6 times per year.

For the maximum frequency of  $U1$ , we use Rule 3 as if  $S2$  and  $S3$  were mutually exclusive. Adding up the frequencies of  $S2 \sqcap U1$  and  $S3 \sqcap U1$  we thus obtain  $1 + 6 = 7$  times per year. Hence, according the indirect calculations, the frequency of  $U1$  should be between 1 and 7 times per year, which is lower than what was obtained through the direct calculation.

The kind of calculations and comparisons demonstrated here can be performed automatically as the indicator values change. This can be utilized to give a warning in cases where the risk picture is inconsistent. In the above example we saw that a certain discrepancy was detected for both vertices that were checked. In practice it is hardly realistic to expect the values to coincide exactly. It is up to the analysts to decide how much two values must differ in order to count as inconsistent, and what should be the exact criteria for triggering a warning.

## 8 Measuring Confidence

The purpose of the Confidence module is to offer a quantitative measure of confidence in the overall risk picture based on the degree of inconsistency that has been detected. There are a number of ways in which notions of confidence may be estimated. The measure could, for example, be based on the number of nodes where inconsistent likelihood estimates are assigned, or on the average difference between conflicting estimates of the same likelihood, or on some more sophisticated statistical analysis. Furthermore, the change of indicator values over time could be considered. If the risk picture has remained consistent over a period of time where indicator values have changed, this gives greater reason to believe in the correctness of the analysis, and in particular the correctness of the functions from indicators to likelihood values, is correct than if consistency has only been observed with one set of indicator values.

Clearly, the degree to which the analysis actually allows consistency to be checked should also be taken into account. Values that are not checked for consistency should count as neither consistent nor inconsistent. In our example above, we are able to check the consistency of likelihood values for  $S3$  and  $U1$ , but not for  $S1$  and  $S2$ . Therefore,  $S1$  and  $S2$  should not contribute to a high confidence value, even if no inconsistency is detected for these vertices.

In cases where two alternative and inconsistent likelihood estimates are obtained for a vertex, the measure of confidence can be employed as an aid to help decide which value is most likely to be correct. This can be done by checking which value gives the highest confidence value. In the example above we obtain two different values for the likelihood of  $S3$  from the indirect and the direct calculations. When calculating the likelihood of  $U1$  indirectly from the likelihood of  $S2$  and  $S3$ , we therefore need to decide which value to use for  $S3$ . Clearly, if we choose a wrong likelihood value for  $S3$  we may also get a wrong likelihood value for  $U1$ , possibly resulting in inconsistent estimates also for  $U1$ , and thus a lower overall confidence value. We should therefore choose the

value that gives the highest confidence value, unless we have other reasons to believe that this value is wrong. After deciding which of two conflicting values is assumed to be most correct, the next step will then be to make the necessary corrections in order to bring the assumed wrong value in line with the correct one, for example by redefining a function for calculating likelihood from key indicators.

A suitable definition of confidence value based on the degree of inconsistency may serve not only as a measure of the assumed correctness of the risk picture, but also as an aid in improving the analysis. How to find the most suitable ways of measuring the degree of inconsistency and the confidence value is a interesting research question in its own right, that we will not pursue further in this paper.

## 9 Related Work

For demonstrating the approach presented in this paper, we have chosen to use CORAS for threat modeling and assignment of quantitative likelihood, consequence and risk levels. However, the approach is generic in the sense that other languages and modeling techniques may also be employed. As we have seen, a suitable language needs to be flexible with respect to annotations of likelihood values and be able to capture inconsistent likelihood estimates, as this allows us to uncover weaknesses of the subjective estimates made by the analysts. We now present some related work, with a particular view on these aspects.

Fault tree analysis (FTA) [IEC90] and related techniques like attack tree analysis [Sch99, MO05] are often used to obtain the likelihood of an unwanted incident in the context of risk analysis. In fault tree analysis, the top vertex represents an event/fault that is decomposed into intermediate and leaf vertices by the use of logical operators. The likelihood of the top vertex is calculated from the likelihood of the leaf vertices, which are assumed to be independent, as well as the operators used to compose vertices. Attack trees are much like fault trees, but focus on the attacks that a system may be exposed to. Moreover, attack trees allow also other values than probability to be assigned to the vertices, for example the cost of an attack, or qualitative statements such as “possible” or “impossible”. As values are assigned only to the leaf vertices by the analysts, there is no possibility of assigning inconsistent values for fault trees or attack trees. In addition, likelihood values are only assigned to vertices, and not to the relations between vertices. From a methodological perspective this means that we cannot define a probability function for a relation from the indicators identified for that relation independently from the related vertices.

Bayesian networks [BG07] are directed acyclic graphs that may be used to represent knowledge and to make quantitative assessments about an uncertain domain, and can therefore be employed in risk assessment. Nodes represent random variables, while edges between nodes represent probabilistic dependencies between variables. A Bayesian network can be used to compute the joint probability distribution over a set of random variables. Like fault trees and attack trees, probabilities are not assigned to the edges of Bayesian networks. Instead, each node is decorated with parameters that for each node give its conditional probabilities, where the conditions represent the state of its parent nodes. The underlying mathematical model of Bayesian networks is

more complicated than that of CORAS diagrams, but also more powerful. Rather than capturing inconsistent likelihood estimates, use of Bayesian networks usually focus on updating likelihood values based on new evidence. Fenton et al [EKN02, EN04] uses Bayesian networks to address the problem of quantifying likelihood based on different types of evidence, and demonstrate how their approach can be applied to assess the frequency of defects in software or components.

Phillips and Swiler [PS98] present a method for risk analysis of computer networks based on attack graphs. An attack graph is not produced directly by the analysis team, but generated automatically from configuration files containing information about the network, attacker profiles containing information about the assumed attacker's capabilities, and attack templates containing information about known generic attacks. Nodes in the attack graph represent attack states (effects of the attack so far), while edges represent a change of state caused by the attacker. Each edge is assigned a weight estimate representing a success probability or some other measure, such as average time to succeed or effort level for the attacker. Multiple weights may be assigned to edges. However, these are not intended to capture inconsistent estimates. Instead they represent potentially conflicting criteria, for example that the attacker wishes to minimize both cost and probability of detection. From the attack trees various kinds of analysis may be performed, such as finding a set of low-cost attack paths or cost-effective defenses. The approach presented in [PS98] does not address the question of inconsistency or confidence in the analysis. It is also less generic than the one we propose, as it is specially tailored to analysis of computer networks, with no emphasis on human behavior.

Closely related to what we call key indicators, the field of IT Security metrics provides an approach to measuring information security. The NIST Performance Measurement Guide for Information Security [CSS<sup>+</sup>08] aims to assist in the development, selection, and implementation of suitable measures to this end. It also provides a number of candidate measures, for example "Percentage of information system security personnel that have received security training" or "Percentage of individuals screened before being granted access to organizational information and information systems". Such measures are suitable candidates for key indicators. Unlike the work we have presented, the approach taken in [CSS<sup>+</sup>08] does not necessarily aim to establish explicit frequency, consequence, and risk levels from the identified set of measures.

An interesting approach to the uncertainty involved in risk analysis based on subjective estimates is taken in [JBK04], which explains the use of subjective logic in risk analysis. Subjective logic [Jos07] is a probabilistic logic that explicitly takes uncertainty about probability values into account. For example, it is possible to calculate to what degree an actor believes a system will work based on the actor's belief about the subsystems. In [JBK04], subjective beliefs and uncertainty about threats and vulnerabilities are used as input parameters to the analysis, allowing the uncertainty associated with the result of the analysis to be explicitly represented.

## 10 Conclusion

We have presented a vision and an approach for risk monitoring where risk values are calculated from measurable key indicators. The resulting risk picture is dynamic in the

sense that risk values are automatically updated as soon as the indicators change. This means that we get a risk picture that remains valid over a period of time rather than representing a snapshot. Moreover, it allows us to consider not only the actual risk levels at a given point in time, but also to analyze trends. For example, if a risk level has been steadily increasing over time, this might suggest that mitigating measures should be considered even if the current risk level is lower than the acceptance threshold. We have demonstrated the approach on the CORAS method, but the same ideas can be used for other risk modeling languages. We claim however that CORAS is particularly suitable due to its flexibility with respect to likelihood annotations.

The approach allows the internal consistency of the risk picture to be assessed in order to reveal weaknesses and issues that need to be reconsidered. A notion of confidence calculated from the degree of inconsistency found in a risk picture has also been proposed in order to assess to what degree the risk picture may be assumed to be correct. This is important because subjective judgment and estimates play a major role in the approach. The aim is not to eliminate the need for such subjective judgment, which would be unrealistic, but to provide support for making the judgment and evaluating its result. Clearly, defining functions from key indicators to likelihood and consequence values based on the subjective judgment of experts will be a major challenge. As noted in [Vos08], eliciting from expert opinion has a number of pitfalls and requires great care, but there are techniques for avoiding the pitfalls. Providing tailored guidelines and methods for defining the necessary functions from key indicators to likelihood and consequence values is an interesting topic for further research.

When making decisions that depend on risks, having a list of potential risks is not enough. We also need to understand how high the risks are. The work presented here is a step towards the goal of obtaining such an understanding.

**Acknowledgements.** The research leading to these results has been funded from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement FP7-216917.

## References

- [BDS08] Brændeland, G., Dahl, H., Stølen, K.: A modular approach to the modelling and analysis of risk scenarios with mutual dependencies. Technical Report A8360, SINTEF ICT (2008)
- [BG07] Ben-Gal, I.: Bayesian networks. In: Ruggeri, F., Kenett, R.S., Faltin, F.W. (eds.) *Encyclopedia of Statistics in Quality and Reliability*. John Wiley & Sons, Chichester (2007)
- [CSS<sup>+</sup>08] Chew, E., Swanson, M., Stine, K., Bartol, N., Brown, A., Robinson, W.: *Performance measurement guide for information security (NIST Special Publication 800-55 revision 1)*. Technical report, National Institute of Standards and Technology (2008)
- [dBHL<sup>+</sup>07] den Braber, F., Hogganvik, I., Lund, M.S., Stølen, K., Vraalsen, F.: Model-based security analysis in seven steps — a guided tour to the CORAS method. *BT Technology Journal* 25(1), 101–117 (2007)
- [FKN02] Fenton, N., Krause, P., Neil, M.: Software measurement: uncertainty and causal modeling. *IEEE Software* 19(4), 116–122 (2002)

- [FN04] Fenton, N., Neil, M.: Combining evidence in risk analysis using bayesian networks. Agena White Paper W0704/01 (2004)
- [IEC90] IEC. IEC 61025, Fault Tree Analysis (1990)
- [JBK04] Jøsang, A., Bradley, D., Knapskog, S.J.: Belief-based risk analysis. In: Proceedings of the Australasian Information Security Workshop (AISW). Conferences in Research and Practice in Information Technology (CRPIT), vol. 32, pp. 63–68. Australian Computer Society (2004)
- [Jøs07] Jøsang, A.: Probabilistic logic under uncertainty. In: Proceedings of Thirteenth Computing: The Australasian Theory Symposium (CATS). Conferences in Research and Practice in Information Technology (CRPIT), vol. 65, pp. 101–110. Australian Computer Society (2007)
- [MO05] Mauw, S., Oostdijk, M.: Foundations of attack trees. In: Won, D.H., Kim, S. (eds.) ICISC 2005. LNCS, vol. 3935, pp. 186–198. Springer, Heidelberg (2006)
- [PS98] Phillips, C., Swiler, L.P.: A graph-based system for network-vulnerability analysis. In: Proceedings of the 1998 workshop on new security paradigms, pp. 71–79. ACM, New York (1998)
- [Sch99] Schneier, B.: Attack trees: Modeling security threats. *Dr. Dobbs Journal* 24(12), 21–29 (1999)
- [Vos08] Vose, D.: Risk Analysis. A quantitative guide, 3rd edn. John Wiley & Sons, Chichester (2008)

# A Risk Based Approach to Limit the Effects of Covert Channels for Internet Sensor Data Aggregators for Sensor Privacy

Camilo H. Viecco and L. Jean Camp

School of Informatics, Indiana University,  
901 E. 10th St., Bloomington, IN 47408, USA

**Abstract.** Effective defense against Internet threats requires data on global real time network status. Internet sensor networks provide such real time network data. However, an organization that participates in a sensor network risks providing a covert channel to attackers if that organization's sensor can be identified. While there is benefit for every party when any individual participates in such sensor deployments, there are perverse incentives against individual participation. As a result, Internet sensor networks currently provide limited data. Ensuring anonymity of individual sensors can decrease the risk of participating in a sensor network without limiting data provision.

Two contributions are made in this paper. The first is an anonymity mechanism to defeat injection attacks. This defense mechanism is based on economics rather than classic cryptographic protocols. The second builds on the foundations created by the first. It is the a proposal for randomized sampling of correlated sensory inputs to asymmetrically increase the cost of sensor identification for attackers without significantly reducing the quality of the published data.

## 1 Introduction

The problem of sensor anonymity is derived from a need to share data. Our solution is constructed upon a foundation of network protocol analysis, information theory, and economics, rather than cryptographic assurances of anonymity. We begin by describing Internet sensor networks, then provide a brief overview of previous work on anonymity-enhancing network protocols. We also define the limitations of previous approaches and illustrate the advantages of the proposed approach.

After this high level introduction, we focus on probe attacks for various classes of sensor networks. This includes a high level description of how attackers use probe networks to obtain covert channels.

The third major section details our proposed approach. We conclude that in the daily operation of sensor networks, economic incentives, and information theoretic defenses that increase the cost to attackers can create an effective defense.

## 2 Incentives and Internet Security

That incentives are a critical issue in economics of information security has been well-documented. In this section we briefly address particular findings on incentives and information sharing in economics of security that are applicable to the question at hand. For a full bibliography on economics of security, please see <http://infosecon.net/workshop/bibliography.php>.

At the individual level, incentives for investment in security are not adequate for socially optimal investment in security. [32] There are negative externalities in the economics of security, meaning that the cost of lack in investment in security is borne not just by the party who can choose to invest but by all participants who bear the cost of spam and botnets. In contrast, there are positive externalities to participation in Internet sensor networks, since all recipients of the information profit not just the participants.

Incentives for investments in security by firms are particularly hindered by a lack of information on the nature of the risks. Despite the number of surveys on the issue of network exploits and system vulnerabilities, there exists considerable gaps in public knowledge of information security. [21] There is even some question about the ability of firms to evaluation the cost of their own intrusions, as similar intrusions result in damage estimates that vary by orders of magnitude. [12]

In terms of information sharing about risks, even at the individual level the risks to security [4] and privacy [26] are not visible. At the organizational level there are incentives to share information, particularly about breaches. However, this incentive requires a closed set of participants who share some commonality, as is the case with an industry-specific ISAC. This information sharing increases investment in security among participants. [11]. These incentives vary by industry, with more concentrated industries and industries with high margin products generally having less incentive to share information and invest in security. [9] In fact, public disclosure laws have encouraged not only information sharing but also investment in security by companies operating under those requirements. [16]

In summary, there is a very real need for information on the state of network security. It is critical that both institutions and organizations have improved data on the state of network vulnerability. Even with that information, investment in security may arguably be inadequate. But without that information, it is not possible for individuals or organizations to make fully informed risk decisions. Sensor networks are specific application of economics of security, as these are inherently information-sharing networks that produce a common value. Thus it is feasible to consider the incentives and disincentives to participation in sensor networks both from the perspective of an attacker (or malicious agent) and a defender (or anonymous participant).

## 3 Internet Sensor Networks

Attacks can be roughly categorized into two groups according to their targeting strategy: directed attacks and undirected attacks. Directed attacks or targets of

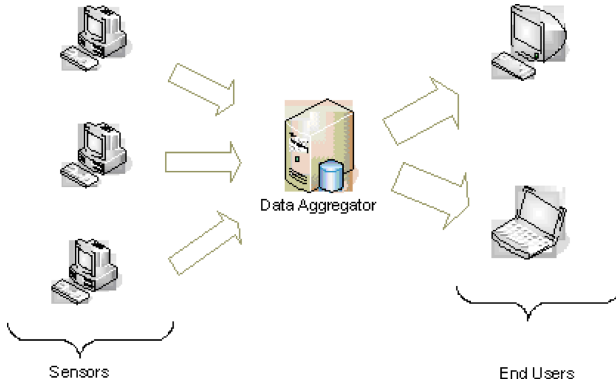


Fig. 1. Data Flow For A Data Aggregation Service

choice occur when attackers purposely mount an attack on a previously identified and selected organization. Undirected attacks or targets of opportunity occur when attackers are searching for some class of resource in order to exploit it. In undirected attacks, the location of such resource is of minor importance. Organizations usually have very different approaches to defending against these two types of attacks; thus, being able to distinguish them is extremely useful.

Differentiating between directed and undirected attacks requires information about the global state of the Internet as close as possible to real time. Data on network status enables administrators to classify threats as directed or undirected, and thus choose an appropriate defense. In addition, by indicating the breadth of an attack, the victim can identify possible allies and collaborative sources of information. Global data enables administrators to better respond to abnormal behavior in their own systems. However, as each network administrator can only know the status of the network under his/her control, data sharing is required to produce a global view. Cooke et al.[29] show evidence that distributed data sharing is inadequate as different address blocks observe different traffic patterns. Thus, even a large aperture sensor is inadequate for knowledge of the network state if it is located in a continuous address block. Widespread sensor placement is required to have a representative sample of the global Internet. It is not a surprise that multiple data aggregation services have emerged to provide such global view.

Aggregation services collect, transform and publish some summary of the information locally gathered by the sensors. Sensors are individual sources of local network status such as honeypots or IDS. Examples of such services/systems include the Internet Storm Center (ISC or Dshield)[30], the Worminator[14], Neti@home[27], myNetwatchman[17], CAIDA[8], the University of Michigan Internet Motion Sensor[31], and the US Department of Homeland Security PRE-DICT system[18]. All these aggregation services work in a similar manner: data



from sensors are collected, filtered and published at a predefined rate<sup>1</sup>. The rates vary between services from one hour to twenty-four hours. The scope of publication also varies, with Dshield publishing the least detailed data to the public at large and the UM Internet Motion Sensor publishing detailed data only to its members. These observations indicate the understanding of the existence of a trade-off between the value and the risk of data availability. There is a concern that more public and detailed data may be more useful to attackers than to defenders. Effective anonymization of data sources can mitigate this trade-off between empowering defenders and enabling attackers.

The relationships between sensors, aggregation services and users of the service are summarized in Figure 1.

### 3.1 Previous Work

Maintaining source anonymity of widely published data has been a problem of interest in politics for several centuries<sup>2</sup>. The problem of measuring the efficacy of anonymization methods has two recent theoretical and practical contributions for measuring the efficacy of anonymization are important to this work. The first comes from Latanaya Sweeney [28,29], who not only reintroduced and analyzed the problem of cross-data identification, but also provided a solution for static data sets called k-anonymity. The second contribution comes from Serjantov and Danezis [25] who redefined the concept of ‘anonymity set’ in a more precise and information theoretic manner. Serjantov and Danezis illustrated that several methods presumed to yield a high anonymity set provided much less anonymity than previously thought. While their work is based on mix networks their ideas can be expanded to other anonymity producing methods.

In the network security arena, the first efforts at providing methods for anonymity came from Flegel et al. [3,7]. Their efforts were directed at removing power from system administrators through anonymization of system logs. Minshall [15]; Fan et.al [6]; and Pand and Paxon [19] provided partial solutions to the problem of anonymization of IP addresses on captured packet traces. Slagel et al. [25] focused on the problem of netflow anonymization. Lakshmanan et al. [13] proposed a generic transformation widely applicable to communication headers. Lincoln et al. [20] proposed a structure to enable searching of IDS alerts in order to detect correlations. Unfortunately, with the exception of the packet traces anonymization methods and the works of Lakshmanan et al. [13] and Lincoln [20], the efficacy of the proposed solutions or methodologies have not been tested against data linking. In the case of packet traces, the possibility of cross data linking is made explicit but never analyzed.

<sup>1</sup> Actually, the DHS's PREDICT system would work on base of NDA agreements. It is still unclear of the need for a trust chain for researchers will be a limiter in the use of the data.

<sup>2</sup> Examples of anonymously published political documents include the Federalist Papers, and the translations of ‘The Rights of Man and the Citizen’, which were not welcomed by colonial powers at the end of the 18<sup>th</sup> century.

Bethencourt et al. [2], was the first researcher to illustrate the problems of cross data linking in Internet sensors. The set of proposed solutions does not include measurements, nor does it provide theoretical bounds on the effectiveness of their solutions. This paper complements their work by providing a theoretical framework in which to address the problem of probe attacks as well as giving potential solutions to a system with the parameters as Dshield.

Clayton et al. [5] makes a good introduction on the fallacies of some data anonymization systems. In particular, they conclude that: "... no operation concerning a pseudonym should have an observable side effect that could leak the identity of the user...". Internet sensor networks, the domain of interest, are designed to show side effects. Yet the identity of Internet sensors (the IP address), should remain hidden.

Another area of interest is the privacy preserving data mining. In particular, the work of Agrawal et al. [1], and Brickell and Shmantikov [26]. Their research is targeted on effectively anonymizing the sensors from data miners by using cryptographic or data perturbation techniques. We will explain what differentiates our work from previous work in section 1.4. We will provide details of the problem space, including the attacks models and trust assumptions, in section 3.2.

### 3.2 Defining the Problem

Our model assumes that the adversary has very little control over the network infrastructure, but does have complete control on many end points. We assume that the aggregation service is trusted by all the sensors, in that the aggregation service will not reveal the identity of the sensors. We assume that there is some mechanism to ensure that the communication channel between the sensors and the aggregation service is protected against traffic analysis. We assume that the aggregation service can uniquely identify any sensor with whom it has previously interacted. We assume that full aggregated data are available to the attacker (he/she belongs to the data sharing consortia), and that an attacker has control over some, but not a significant part of the sensors. We further assume that the sending of probes has a very small yet non-zero cost to the attacker. The problem that we are trying to solve is: Is there a way to make the probe sensor identification of a large portion of the sensors economically unfeasible? Can we provide a high lower bound on this cost? Further, Can we measure how much our data output changes when different mitigation mechanisms are applied? This last question will only be analyzed for a Dshield like system.

The assumption that an attacker is able to compromise multiple end points but not as likely to compromise infrastructure nodes is simply the recognition of botnets [23]. In our trust model, we trust the aggregation service, but do not trust the other entities that are also receiving the data from the sensors. This is consistent not only with botnets, but also with a grayhat adversary or adversaries that are competitors in other arenas.

One of the interesting elements of this problem is that attackers use the infrastructure, i.e. the reports of the sensor networks, to attack the infrastructure, the location and accuracy of the sensor network. This particular study focuses on

adversaries that cannot control or observe how the information passes through the network, rather focuses on adversaries that take advantage of the implicit feedback loop generated by the process of publishing the data.

The key differentiators of the sensor network anonymization probe are: (i) the data are not static, data is periodically added to the output (ii) the data provided by the aggregator are available to the attacker, and (iii) the defender cannot distinguish 'a priori' probe data from bad injected data.

### 3.3 Comparison with Previous Approaches

With all the assumptions detailed above, it is reasonable to believe that this problem can be solved by applying previously published anonymization techniques. In this section we explain why some general techniques fail to address our problem.

Data filtering may seem like an obvious approach. The problem with data filtering is that abnormal network status data injected by attackers cannot be distinguished from abnormal network data due to non-probing attackers.

Mix networks or onion routing cannot be used as a defense mechanism against probe attacks (data injection) as these are designed to address a different problem. Mix networks and onion routing provide unlikable communication channels across untrustworthy communication intermediate peers that are trying to determine who is communicating with whom. In our solution and model, this part of the problem is assumed to be solved potentially by some implementation of these mechanisms such as Tor[27]. Further, our problem statement differs from anonymous communication problems in that our adversary has very limited control of the infrastructure, yet still controls many end points.

Sweeney's [28,29] emphasizes the use of k-anonymity only for static data sets. The process of re-identification of datasets is usually done with the use of external data utilized for cross data linking. For data that increases over time where the attacker has some control, another method can be used: the use of probe response attacks. The possibility of such attacks in the Internet has been known in the literature [19] but it was not until the work of Bethencourt et al. [2] that an algorithm and simulations were published. Bethencourt et al. demonstrated the problem by showing how simulations allow easy discovery of sensors of the ISC [30]. In this paper, we generalize the costs for such identification procedures for any aggregation service in addition to providing guidance to mitigation mechanisms. The procedure we introduce increases the cost for the attacker while minimizing the distortion of the data released by the aggregation service.

The proposal of Agrawal et al. [1] consists of adding a random variable to the sensor data to effectively perturb the data output. If the random variable has a very large variance, this method requires a large number of inputs to effectively approach the original distribution. If the variance is small, the attacker need only to generate data outside the variance to create a reliably detectable signal.

The work of Lincoln [20] includes several techniques for anonymization and related defense mechanisms. The method they propose against probe response

attacks is the use of randomized delay alert correlation, with the time stamp field scrubbed. This method cannot be reasonably used for our purposes, as data sharing for operational use requires near real time latency. Further, strategic (long term) use requires timestamps with at least a one day resolution.

The work of Brickell and Shmantikov[26] uses cryptographic techniques to unlink data thus protecting individuals from releasing their identity to data miners. However this work does not take into account the possibility that the data being reported can be influenced by the party that is trying to identify the identity of the data sources.

The approaches suggested by Bethencourt et al. [2], in particular the sampling of data outputs, appears to be a good compromise. In particular, Bethencourt uses economic incentives to prevent ‘marking’ of packets. The problem with this approach is that sampling is done on a per sensor level, after data have been collected. This approach does not increase the signal to noise ratio for the attacker. This approach does not work if we assume attackers with access to large botnets, as the defense mechanism leaves the attack trivially parallelizable.

All of the previously suggested techniques address the problem after the data have been aggregated. In economic terms, these post-collection sampling mechanisms provide more advantage to the attackers than the defenders. Post collection data transformation are more expensive for the aggregator than injection for the attacker, thus creating a systematic asymmetry. The approach presented here advantages the defenders by utilizing the ability to apply sampling at different dimensions and in different levels at event recording time. Thus attackers must synchronize their injected signals in all the possible filtering dimensions. The result is an economic disadvantage for the attacker, as described in more detail in the following pages.

## 4 Probe Attacks and Internet Sensor Networks

Internet sensor networks are the data source for Internet status aggregation services. Aggregation serves two functions: It centralizes data publishing, and enables limited anonymization of the sensors. Sensor anonymization is a fundamental requirement for the contributors as well as the quality of the aggregate data. An attacker who can identify the sensors will be able to: (i) hide attacks (hide or slow worm spread), (ii) hide a directed attack to an organization, or (iii) completely distort the quality of the exported data, thus making the data sharing effort useless.

Anonymization is so important, that despite economic benefits to data sharing [10], sharing detailed information security data is usually highly limited. Organizations that share internal data include the HoneyNet Alliance [22] and the business sector-based ISAC structure in the US. But even within those groups, data are aggregated, filtered and thus transformed before release. (The HoneyNet Alliance is a notable exception to this rule. Sensor anonymity is not an issue in the HoneyNet Alliance as the lifespan of honeypots is usually limited to a few intrusions). Many of the current data sources include sensors that are not

easily relocated, such as Darknets. For others, the shared information is usually reduced to summaries of data for example as with the REN-ISAC [24].

### 4.1 Probe Internet Sensor Attacks

Probe sensor attacks use the feedback channel implicitly provided by the data compilation and aggregation. Since the publication phase cannot distinguish “good” users from “bad” users, a malicious user can send traffic into potential sensors to try to observe the abnormal signals in the output of the data aggregation. See Figure 2.

The most generalized statement of the problem from the attacker's perspective is: “Determine the parameters of a box with some controllable inputs and some observable outputs”. However the sensor identification problem differs from most system identification methods because our system has many inputs and outputs, and is generally non-linear. The attacker's objective is to estimate the sampling function used to collect the data from the Internet. The function's secret parameters are the true location of the sensors, as the remaining parameters must be published in order to make sense of the published data.

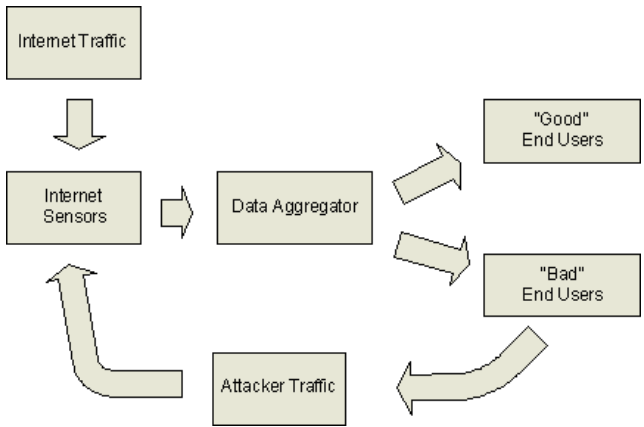


Fig. 2. Data Flow For Probe Response Attacks

The costs associated with running sensor identification attacks can be explained by running time and bandwidth costs. We will discuss two attack algorithms: a brute force approach and an N-ary recursive approach. These are analyzed in terms of their “running time”. This parameter is used to estimate the cost for an attacker. The running time of an algorithm describes a bound on the number of operations needed to complete the algorithm. The cost for each algorithm is expressed in terms of the needed bandwidth required for its operation.

**Linear (Brute Force) Algorithm.** This algorithm essentially iterates through each of the possible sensors to determine if it is a sensor or not. The algorithm is expressed below:

1. For each possible location
2. Estimate the number of sensors in the current selected location
3. if number of sensors is zero
4. then discard location
5. else location is a sensor

The running time of this algorithm is:  $U*K$ . Where  $U$  is the size of the search space and  $K$  is the number of iterations needed to determine whether a sensor has been located. In this case, a partition of size one. The bandwidth cost per iteration is  $P$ , where  $P$  is the number of packets required to generate a readable signal in the aggregate data. The total cost for such algorithm is calculated by multiplying the running time by the per iteration cost:

$$\text{Total cost} = \text{running time} * \text{iteration cost} = U*K*P.$$

This algorithm has a minimal cost, but also has a linear running time. A linear running time is unfeasible for large sensor spaces, such as the Internet.

**N-ary Search Algorithm.** Another way to approach sensor identification is to use a divide and conquer approach. In this algorithm (based on the one published by Bethencourt et al. [2]), the possible search space is partitioned at each iteration. A partition can be discarded if it contains no sensors, or the partition is of size one, meaning that the location of the sensor has been discovered.

1. Make the set of non-empty partitions={all the search space}.
2. while the set of non-empty partitions is not empty do
3. Extract one of the element from the set of the non-empty partitions. Name it  $x$ .
4. Partition  $x$  up to  $N$  partitions.
5. For each of the subpartitions of  $x$  do:
6. Estimate the number of elements in it.
7. if the number of elements in the subpartition is zero
8. then discard the subpartition.
9. else if size of subpartition is equal to one
10. then sensor has been located
11. else insert the subpartition into the set of non-empty partitions

The maximum running time of the algorithm is  $O((\log U)*S*K)$ . Where:  $U$  is the size of the search space;  $S$  is the number of sensors; and  $K$  in the number of iterations needed to estimate the number of sensors in a partition. The expected running time assuming a uniform distribution of the sensors is also  $O((\log U)*S*K)$ . The change from a linear  $U$  dependency to a logarithmic  $U$  dependency is due to the comparison in step 7. Once a portion of the search

space has been determined without interest, it can be safely disregarded. Thus most of the research has evolved on making this comparison to zero unreliable[2]. The side effect of this algorithm's reduction in time is an increase in resources needed. In particular, the cost of each iteration is the partition size times P. As the maximum partition size is  $U/N$ , where N is the maximum number of partitions, the cost per iteration is bounded by  $U/N * P$ .

The total cost is then:  $O(\log U * S * K) * \text{cost\_per\_iteration} \leq O(\log U * S * K * P * U / N)$ .

## 5 A Risk Based Approach

The previous analysis assumed it is possible to detect a specialized signal injected into the system, by injecting some special packets. While there is no proof that this can be done with 100% certainty, it can be proved that retrieving a signal over time can be done with arbitrary precision given some very lax conditions (This proof is on the appendix). Given this fact, data aggregator designers must optimize the expense, not the possibility of an attack. Like a work factor in cryptography, solutions must have very large bounds. In our analysis, we have assumed that the sensor location is fixed for the duration of the sensor attack. This assumption approximates current practices and limits the usage of the equations, but provides useful guidance for future deployments. This is also the worst case scenario.

The previous equations show dependencies on:

- U: the size of the potential sensor identifications, ie. the a priori size of the anonymity set;
- S: the number of sensors in the aggregation service (S);
- K: the number of iterations required to make a decision, or the number of iterations required to reliably detect the attacker's signal.
- P: the number of packets required per iteration to generate a readable signal.
- N ( in the N-ary case), the number of partitions that we can make per iteration or the number of orthogonal signals that we can inject into the system (with the assumption that the costs are the same).

Only two parameters can be controlled by the aggregator service: K and P. The design goal for data aggregator is to implement aggregation methodologies that increase these two values for the attacker while having a smaller effect on the overall aggregated data (This is in lieu of database perturbation methods). Again the key is to measure how well each possible implementation affects both the attacker and the defender.

### 5.1 P: Noise and Sensitivity

The P parameter is the minimum amount of effort required to insert a detectable signal in the published data. This value is directly related to the sensitivity of the system and the noise level of the system. For a linear system (such as

the D-shield),  $P$  needs to be chosen depending on the average value and the deviation of the undisturbed output.  $P$  is also related to the resolution of the output channel, the set of possible output values for each value in the dataset. In general, increasing the size of  $P$  reduces the sensitivity of the output or increases the signal to noise ratio.

$P$  can also be thought as an economic disincentive value. Increasing  $P$  increases the marginal costs for attackers as more resources are required to extract the identity of any sensor. The precise value and effect on attackers depends not only on  $P$ , but also on the problem specific costs per probe. In the case of simple network probes, this cost is almost negligible given the possibility of large botnets [23]. For other types of monitors where more interaction is required, this approach might yield the best results.

Another advantage of  $P$  is that it is easy for the aggregator to calculate. The other parameter,  $K$ , is harder to estimate, thus, assumptions about its efficacy must be carefully detailed by both designers and deployers of Internet sensors.

## 5.2 K: Uncertainty and Entropy

The  $K$  parameter represents a measure of the amount of information that can be extracted from the published data per each interaction.  $K$  is an information theoretic limit on the properties of the published data which depends on the interaction of the aggregation service with the sensors. In particular,  $K$  for the  $n$ -ary algorithm is the number of iterations needed to determine estimate with arbitrary precision that there are no sensors present in a subset. Augmenting the  $K$  parameter does not imply an increased cost in resources for the attacker but an increased cost in time. An increase in  $K$  requires a longer running time that cannot be compensated by more resources (compromised systems). For low interaction systems, where the number of sensors is sufficiently large, the immediate way to generate an increased  $K$  is the use of sensor sampling at the aggregator level. For systems which provide richer and more sensitive data, there is no clear way to achieve anonymization while preserving the probabilistic properties of the data. As there is more entropy in the data and this a large place for attacks to put unique ‘tokens’ in the data. Using sampling at the sensor level, the number of iterations required to determine the presence of a sensor with precision  $r$  when the per sampling rate is  $p$  is given by:  $\log(1 - r) / \log(1 - p)$ . Notice that this value is independent on the how the markings are done or the independent cost per probe.

Another possible way to increase the cost is not to directly increase  $P$  or  $K$ , but to increase the communication effort needed to potentially scan a host. If sufficient communication overhead is placed on the attacker then the “free” bandwidth and cycles of the compromised machines stops being “free”. However this is beyond the control of the aggregator.

It is important to emphasize that it is impossible to prevent the use of the system output as a verification oracle. The goal of the techniques and methods proposed here is to significantly increase the cost of using the system as a verification oracle for multiple systems simultaneously. Confirmation attacks are still



possible, but the use of the attacks to explore the address space is no longer feasible.

### 5.3 An Example with Dshield

Previously discussed is the need to increase the values of  $P$  and  $K$  as much as possible to make the cost or the time required for an attacker to be sufficiently large. In this section we will discuss mechanisms for a well documented and understood aggregator service: Dshield.

**Dshield Operation.** Dshield collects data about unexpected connection attempts to computers. Its sensors are end hosts' firewall logs. These logs are given voluntarily to Dshield by the internet community. Dshield aggregates such logs and reports the number of connection attempts per port every hour. Dshield also reports the number of hosts and the number of sensors that observed such behavior. Dshield was the first aggregation service studied for probe attackers by Bethencourt et al. [2]. That work described two types of defenses against probe attacks: social and technical. Social methods include pricing the published data and the use of private reports. But pricing the data would make the data less useful and the use of private reports can only help if there no attackers are also sensors.

Technical measures suggested include: per packet sampling, use of top lists, scan prevention and Delayed reporting. However all of these methods have inherent problems. Per packet sampling generates an increase in  $P$ , but does not address the parallelization of the attack. Top lists changes the nature of the reported data. Scan prevention such as the use of IPv6 address space would make the system not useful. Delayed reporting is problematic as late data is of no good for most uses and in fact probe attack efficiency is reduced by only a constant.

We believe that other methods can be more effective at providing the same level of protection to the sensors. In particular the increase in  $K$  is not discussed and might be one of the most powerful incentives to prevent such attacks.

**Increasing  $P$ .** The easiest way to increase  $P$  is to use of per packet sampling. By selecting a packet to be reported with probability  $p$  the attacker must select its reliability measure  $r$  (probability of not detecting a sensor) and then he/she needs to send at least  $\log(1-r)/\log(1-p)$  packets per iteration per destination.

There are two problems with this approach: First while sampling augments the amount of packets required to detect the signal, it also reduces the noise level and thus some channels that were previously unusable due to noise become available. Second this sampling technique does not over count the packets. Therefore an attacker can use this information to determine an upper bound on sensors in a partition. The attacker can end probing on a subpartition when that bound is reached.

A potentially better method is the use of randomized sampling. At each period each sensor selects a probability between  $p_1$  and  $p_2$  (with uniform distribution

between these two values). By using this method three things happen. First, the attacker must use the lowest probability to guarantee that his signal is observed while the sensors average probability is  $(p_1+p_2)/2$ . Second, this introduces some noise factor. Third, probes can be over counted, this overcounting prevents the attacker from discarding any sub partitions when thresholds are reached. The advantage is the information asymmetry of the method. There is a difference between the guaranteed probability of selection and the expected probability of selection. In other words, this method disturbs the data more effectively for attackers than for defenders.

Other approaches include the use of buckets of defined sizes to group data or limiting the resolution of the output signal. Resolution is decreased by limiting the number of significant digits of the output. However the effect of these methods is similar to the simple per packet sampling.

**Increasing K.** This section provides multiple mechanisms to increase K. Recall K is the information theoretic limit on the cost function. One way to potentially increase K is also to use sampling, but at the sensor level. At each time interval, the aggregation service will select the logs from some sensor to be added to the aggregate list with some probability p. By using this sampling, an attacker signal for each sensor would be lost at each interval with probability (1-p) no matter what type of signal he/she introduced.

Another way to increase K is the use of data correlations. As Dshield is designed to detect automated threats, we can use certain domain specific knowledge about such threats. Data are uncorrelated in the source IP address. Data are also uncorrelated in the time domain. With this in mind, assume that the lower X bits of IP address space are uniformly distributed and use them to sample. By sampling on one of these bits, an attacker using only one compromised machine has a 50% change of not being reported, independent of the number of probes sent to the sensor. Time is the other possible correlation dimension we can use. The time sampling mechanism could select randomly only even seconds or only on the first half hour ( or every uneven packet ). This would force the attacker to not only use more resources but also to spread them in a more uniform distribution. This requires synchronization among all the systems used by the attacker to launch the attack.

In general, the use of data correlations does not directly increase K, but provides a large disincentive to try to determine the location of a sensor. The probability of observing the output, given an attacker that can generate different packets that match each of our selection dimensions is given by:  $P_{randomselection} + (n - 1)/N$

Where

- $P_{randomselection}$  is the base probability of selecting any one random packet,
- $n$  is the number of probes sent by the attacker that are in different dimension
- $N$  is the total number of possible selections.

Another option is to use a Markov chain to select whether a sensor reports data back to the aggregate sensor. While this does not provide extra protection

it increases the complexity of the attack. The order of the running time remain the same, but the attacker is forced to store more state. Specifically attackers must interleave the sampling of data and cannot use depth first attacks. Markov chains and data correlations are examples of the use of information asymmetry.

The requirement of state data makes the attack more expensive while not changing the accuracy of the collected data.

**Limitations of the methods.** All methods discussed in the increase of  $K$  and  $P$  have two potential problems. The first is that the data quality of the collection system is decreased. However, assuming the distributions are uniform in the dimensions selected for sampling, adding noise does not change the expectation of the output before and after using the proposed methods. The real problem comes from calculating the expected deviation of this output given each sampling mechanism.

The second potential pitfall is the that the system is more sensitive to rogue sensors. The effects of rogue sensors can be amplified with sampling is the sensor implements the sampling and provide malicious data. However, if the sensors are required to submit sampled sanitized data then abnormal deviations of sampling values can be detected. The sensor system still needs to use other tools to validate the data reported by individual sensors, but this question is out of the scope of this paper.

## 6 Conclusions

Anonymization procedures employed by aggregation services are a unique and important special cases of anonymity. Without proper anonymization, those services vulnerable to injection attacks and reduces the confidence of sensors. The absence of an absolute method to assure anonymity for sensors indicates that economic and information theoretic approaches are needed. We have shown the fastest known algorithm for sensor location and the kinds of mitigation mechanisms that can be put in place. We introduced two parameters that can be used to explain the effectiveness of potential mitigation solutions.

In the particular case of the Dshield we have enumerated the problems of other currently proposed defense mechanisms. We have offered a set of methods that can be used to avoid said enumerated problems such as the use of randomized packet sampling, sensor sampling and correlation sampling. Further we have shown that specificity when describing sampling methodologies is required. Sampling in different spaces generates different dependencies.

Further research is required into the efficacy of leveraging determining information asymmetries. Currently we are working to determine how robust our proposed methods are against rogue sensors. Currently deployed data aggregators must implement defense mechanisms as soon as possible in order to guarantee the accuracy of their data set. Future aggregation services must spend more time in the analysis of the anonymization mechanisms specifically in on how to generate anonymization methods with highest marginal costs for attackers.

## Acknowledgements

This work is sponsored by Indiana University's Advanced Network Management Lab (ANML) and the Institute for Information Infrastructure Protection (I3P) research program.

## References

1. Agrawal, R., Srikant, R.: Privacy preserving data mining. In: ACM SIGMOD. ACM, New York (2000)
2. Bethencourt, J., Franklin, J., Vernon, M.: Mapping internet sensors with probe response attacks. In: Proceedings of the 14 USENIX Symposium. USENIX (August 2005)
3. Biskup, J., Flegel, U.: On pseudonymization of audit data for intrusion detection. In: Federrath, H. (ed.) Designing Privacy Enhancing Technologies. LNCS, vol. 2009, p. 161. Springer, Heidelberg (2001)
4. Camp, L.J.: Reliable, usable signaling to defeat masquerade attacks. In: Workshop on the Economics of Information Security, Cambridge, UK (June 2006)
5. Clayton, R., Danezis, G., Kuhn, M.G.: Real world patterns of failure in anonymity systems. In: Moskowitz, I.S. (ed.) IH 2001. LNCS, vol. 2137, p. 230. Springer, Heidelberg (2001)
6. Fan, J., Xu, J., Ammar, M.H., Moon, S.B.: Prefix-preserving ip address anonymization: measurement-based security evaluation and a new cryptography-based scheme. *Comput. Networks* 46(2), 253–272 (2004)
7. Flegel, U.: Pseudonymizing unix log files. In: Proceedings of the Infrastructure Security Conference (October 2002)
8. C. C. A. for Internet Data Analysis. Caida website (July 2005), <http://www.caida.org>
9. Gal-Or, E., Ghose, A.: The economic consequences of sharing security information. In: Camp, L.J., Lewis, S. (eds.) Economics of Information Security. Advances in Information Security, ch. 8, vol. 12, pp. 95–105. Springer, New York (2004)
10. GalOr, E., Ghose, A.: The economic consequences of sharing security information. In: Workshop on the Economics of Information Security (2003)
11. Gordon, L.A.: An economics perspective on the sharing of information related to security breaches: Concepts and empirical evidence. In: Workshop on the Economics of Information Security, Berkeley, CA, USA (May 2002)
12. Granick, J.: Faking it: Criminal sanctions and the cost of computer intrusions. *I/S A Journal of Law and Policy for the Information Society* (2006)
13. Lakshmanan, L.V., Ng, R.T., Ramesh, G.: To do or not to do: The dilemma of disclosing anonymized data. In: SIGMOD, June 2005, pp. 61–72 (2005)
14. Locasto, M.E., Parekh, J.J., Keromytis, A.D., Stolfo, S.J.: Towards collaborative security and p2p intrusion detection. In: Information Assurance Workshop (June 2005)
15. Minshall, G.: Tcpspriv man page (1997), <http://ita.ee.lbl.gov/html/contrib/tcpspriv.html>
16. Mulligan, D.K.: Information disclosure as a light-weight regulatory mechanism. In: DIMACS Economics of Information Security Workshop (2007)
17. T. myNetWatchman Project. The mynetwatchman website, <http://www.mynetwatchman.com>

18. D. of Homeland Security. Predict system
19. Pand, R., Paxson, V.: A high-level programming environment for packet trace anonymization and transformation. In: SIGCOMM 2003. ACM, New York (2003)
20. Patrick Lincoln, P.P., Shmatikov, V.: Privacy sharing and correlation of security alerts. In: Proc. USENIX Security 2004 (2004)
21. Pfleeger, S.L., Rue, R., Horwitz, J., Balakrishnan, A.: Investing in cyber security: The path to good practice. The RAND Journal (2006)
22. T. H. Project. The honeynet project website
23. T. H. Project. Know your enemy: Tracking botnets, <http://www.honeynet.org/papers/bots/>
24. Research and E. N. ISAC. REN-ISAC monitoring website
25. Serjantov, A., Danezis, G.: Towards an information theoretic metric for anonymity. In: 2002 Privacy Enhancing Technologies Workshop (2002)
26. Shostack, A., Sylverson, P.: What price privacy? In: Camp, L.J., Lewis, S. (eds.) Economics of Information Security. Advances in Information Security, vol. 12, pp. 129–142. Springer, New York (2004)
27. Simpson, C.: Neti@home website (August 2005), [www.neti.gatech.edu](http://www.neti.gatech.edu)
28. Sweeney, L.: Computational Disclosure Control: Aprimer on Data Privacy Protection. PhD thesis, Massachusetts Institute of Technology (June 2001)
29. Sweeney, L.: k-anonymity: A model for protecting privacy. International Journal on Uncertainty Fuzziness and Knowledge-Based Systems 10, 555–570 (2002)
30. Ullrich, J., Consulting, E.: Dshield homepage (August 2005), [www.dshield.org](http://www.dshield.org)
31. M. N. University of Michigan and A. Networks. University of michigan internet motion sensor (2005), <http://ims.eecs.umich.edu>
32. Varian, H.: System reliability and free riding. In: Sadeh, N. (ed.) Proc. of the ICEC 2003, pp. 355–366. ACM Press, New York (2003)

## A Appendix

What we have then is a problem of system identification. These type of problems are very common in the control theory, and our problem though similar has three properties that make them a little bit different: (i) exact knowledge of complexity of the transfer function, (ii) large input space, and (iii) non-zero mean (or median) error. The exact knowledge of the complexity of the transfer function means that we know the exact structure of the system we want to identify, thus the identification task is to generate good approximations for the parameters of that structure. The fact that this structure is known a priori in general reduces the complexity of the identification procedure. Usually identification systems have a relatively small input space of order less than  $10^3$  where in our case we have a large set of inputs, that is all valid Internet end points around  $10^9$ . This means that methods that require large number of input-output probes cannot be used as the space cannot be generated or stored. The non-zero mean error means that some techniques as sum of squares cannot be used directly.

However, the theory and knowledge of system identification procedures can still be used but with some caveats. For our case the critical part is to determine the excitation signals necessary for appropriate identification. These signals must satisfy two conditions: They must cover as much as possible the internal state of the system and they must be detectable (identifiable) in the output. Since an attacker can reach any end point in the system, what really requires study is the detection of the signal.

### A.1 Signal Detection in Discrete Spaces

Discrete signal detection is a known problem in communication systems. In particular, in cases where the transmission channel is linear and the noise is with finite energy and with zero mean (ex. white noise) methods to detect signal are pretty much known. Our case in particular has a finite output space and the signal is also discrete in time. Our function is not linear (in general) and the noise is bounded and has non-zero mean. But even in these case, signal detection with an arbitrary non-zero error is possible under the following circumstances:

1. Ability to excite the channel
2. Noise is i.i.d. (Independent identically distributed) at each time period.
3. System is time invariant.
4. The conditional PDF of the output in the case with no signal is known.
5. For at least one of the possible output values  $y_i$  the conditional probability, given the signal is present is known to differ from the no signal case by at least some known  $\epsilon_i$ . In other words:  $\exists y_i / \|P(y_i | \text{No signal}) - P(y_i | \text{Signal})\| \geq \epsilon_i$

### A.2 Proof, Binary Case

If the output function is binary, from condition (4) we know the signal less distribution  $d$  1 with parameters:  $p_1$  and  $q_1 = 1 - p_1$ . Since this distribution is time invariant (conditions (2), (3)), a sequence of outputs of this distribution

will generate a binomial distribution. This binomial distribution with N trials has:

$$\begin{aligned} \mu_{d1} &= Np_1 \\ \mu_{d1,2} &= \sigma_{d1}^2 = Np_1q_1 \end{aligned} \tag{a.1}$$

From the conditions for identification(condition (5) ) we know that for the signal case we have  $p_2 > p_1$  + (Or the opposite, in which we can rename the output signals). In this case (with signal) the repeated trial would yield to another binomial distribution with:

$$\begin{aligned} \mu_{d2} &= Np_2 \\ \mu_{d2,2} &= \sigma_{d2}^2 = Np_2q_2 \end{aligned} \tag{a.2}$$

Now, we want the error to be least that some  $p_{req}$ . If we set the decision threshold in the midpoint between the two expectations  $t = (Np_1 + Np_2)/2$ . The error of detection is given by the maximum area where the decision threshold gives the opposite value. What is needed is to find an N which the error would be less than that. Using the Tchevycheff's inequality we can say:

$$P|X_{d1} - \mu_{d1} \geq t| \leq \frac{\sigma_{d1}^2}{t^2} \leq p_{req} \tag{a.3}$$

Replacing in equation (4) with values from equation (2) we can come with:

$$\begin{aligned} \frac{Np_1q_1}{(N(p_2-p_1)/2)^2} &\leq p_{req} \\ \frac{4p_1q_1}{N(p_2-p_1)} &\leq p_{req} \end{aligned}$$

Thus:

$$N \geq \frac{p_1q_1}{p_{req}(p_2-p_1)^2}$$

Similarly for the second distribution (with signal) we have:

$$N \geq \frac{p_2q_2}{p_{req}(p_2-p_1)^2}$$

Thus we can find a bound for N that satisfies the requirement for an arbitrary but non-zero error requirement  $p_{req}$  .

### A.3 Arbitrary Case

We can convert an arbitrary function that we know at least some  $\epsilon_i$  into the binary case. And we can use the above proof.


# An Experimental Testbed for Evaluation of Trust and Reputation Systems

Reid Kerr and Robin Cohen

David R. Cheriton School of Computer Science,  
University of Waterloo, Waterloo, Ontario, Canada  
{rckerr,rcohen}@uwaterloo.ca

**Abstract.** To date, trust and reputation systems have often been evaluated using methods of their designers' own devising. Recently, we demonstrated that a number of noteworthy trust and reputation systems could be readily defeated, revealing limitations in their original evaluations. Efforts in the trust and reputation community to develop a testbed have yielded a successful competition platform, ART. This testbed, however, is less suited to general experimentation and evaluation of individual trust and reputation technologies. In this paper, we propose an experimentation and evaluation testbed based directly on that used in our investigations into security vulnerabilities in trust and reputation systems for marketplaces. We demonstrate the advantages of this design, towards the development of more thorough, objective evaluations of trust and reputation systems.

## 1 Introduction

The area of *multiagent systems* is concerned with scenarios where a number of agents (who may be acting on behalf of different users) must interact in order to achieve their goals; often, an agent must depend on other agents in order to achieve its objectives. In such scenarios, trust can be an important issue—an agent's ultimate success may depend on its ability to choose trustworthy agents with which to work. For this reason, *trust and reputation systems* (TRSes)  have received much attention from researchers. Such systems seek to aid agents in selecting dependable partners (or in avoiding undependable ones).

A particular focus for researchers has been on the electronic marketplace scenario, a well-established and important example of a multiagent system. In this setting, agents act as traders, buying and selling amongst one another. The ability to find trustworthy partners is critical to an agent's success, because an untrustworthy agent may deliver an inferior good (or fail to deliver at all), or may not pay for goods purchased. The nature of electronic marketplaces complicates the evaluation of trustworthiness: identity is difficult to establish (because new accounts can be created easily), agents may not engage in repeated transactions

---

<sup>1</sup> For convenience, we use the abbreviation TRS, for 'Trust/Reputation System', in reference to both trust systems and reputation systems.



together (because of the size of the market and the diversity of products), and an agent may have an advantage over another during a transaction (for example, when a buyer must pay in full before a seller ships (or fails to ship) the good).

Along with the multitude of TRS proposals have come a similarly large number of methods to evaluate the proposals. It has been widespread practice for researchers in the field to develop their own testing methods. A common approach has been to conduct simulations using a scenario of the authors' own devising to show the value of their model, often achieved by pitting their own proposal against their implementations of other existing models. There is nothing fundamentally unreasonable about this approach, in the absence of established testing tools. Unfortunately, there have been significant limitations in the evaluations typically used by authors. It is not surprising that testing scenarios used by authors often favour their own work. This is not to suggest any misdeeds on the part of these authors; when designing a system, it is natural to have a particular scenario in mind, and for subsequent tests to reflect that scenario.

Members of the trust and reputation community have invested significant effort in developing the Agent Reputation and Trust (ART) testbed [1]. A primary purpose of ART is to serve as a competition platform, and it serves this purpose well [2]. While ART is a valuable contribution, a number of design choices make it less appropriate for broad use in the experimental evaluation of TRSes. We discuss these issues in Sect. 2.1

Perhaps more important than issues of bias, the evaluation procedures typically used obscure critical problems that have received insufficient attention to date by trust and reputation researchers. Simulations typically make use of agents that are simple or naive in their dishonest activities. For example, many simulations (e.g., [3,4]) are populated by random selections of agents that either always cheat or always behave honestly, or by agents whose cheating is governed by simple probability distributions, where each time step is independent of previous ones. Such simulations ignore the possibility that cheaters might behave in a more sophisticated manner—for example, trying to identify and exploit a specific weakness in the system—providing little comfort to those who might wish to consider these proposals for real-world use. In earlier work [5], we identified a number of common vulnerabilities that might allow attackers to defeat the protection offered by TRSes, and argued the critical importance of security in TRSes. Recently [6], we demonstrated the practicality of such attacks by soundly defeating a number of noteworthy TRS proposals. These results demonstrate the need for more rigorous tests, and more objective tests, of TRSes.

In this paper, we propose a testbed formulation designed to support diverse, flexible experimentation with TRSes, and more thorough, objective evaluation of such systems. This formulation is based directly on the platform used in our experimentation in [6], which was designed to be a general-purpose experimental platform for both of these purposes. Our experience and results have shown it to strongly support both goals.

The design has a number of important advantages making it well suited for its intended purposes, including:

- It models a general marketplace scenario, allowing systems to be tested under realistic conditions. This includes reasonably large marketplace populations, turnover in the agent population, a large number of products/prices, etc.
- It is modular, allowing new TRSes, buying and selling agents, instrumentation, etc., to be added easily.
- It can support a wide range of trust/reputation approaches (for example, both centralized and decentralized models). It does not impose any particular view of trust on agents, nor does it impose a particular protocol or trust representation on agents.
- It allows collusion to be incorporated into agent behaviour.
- It allows individual marketplace ‘components’ to be tested in isolation. For example, it allows the protection a TRS provides buyers from cheating sellers to be evaluated, without being obscured by other potentially irrelevant issues (for example, whether or not sellers are dishonest with one another). In contrast, the success of an agent in ART requires competence in a number of abilities.
- Given the standardized platform, as new agents/TRSes are developed, they can be evaluated against all existing implementations; at the same time, new implementations constitute new tests for all of the existing systems. In this way, a continually improving battery of rigorous tests can be developed, which can be broadly used by researchers to evaluate their work. (The ‘smart’ cheating agents used in [6], for example, constitute an initial set of tests that have proven extremely difficult for existing proposals.)
- Standardization also allows for objective benchmarking, permitting meaningful comparison between systems. Moreover, the availability of components allows for results to be reproduced by other investigators.

We believe this testbed to be a valuable tool in its own right; moreover, we believe it to be an important step towards thorough, objective evaluation of trust and reputation systems.

## 2 Related Work

The majority of TRS proposals applicable to marketplaces have been evaluated using methods of their authors’ own devising. (Subsequently, these systems might appear as comparison data points in later proposals’ own self-devised tests.) Methods used have included mathematical analysis of properties (e.g., [5,7]) and simulation (e.g., [3,4,8]). Both of these approaches are reasonable. The evaluations performed, however, have proven to be problematic. Because each evaluation is different, results presented by different authors are not comparable. The evaluations presented are often quite brief, leading one to question whether the results thoroughly reveal the performance of the systems in question. Indeed, our investigations [6] revealed numerous ways in which the systems cited above can be defeated—issues that were not revealed in the authors’ own analyses. These issues highlight the need for more thorough, objective testing of TRSes, ideally using tools that allow comparison and reproducibility of test results.

## 2.1 The ART Testbed

A standardized, common testing platform can potentially address the issues noted above. The Agent Reputation and Trust (ART) Testbed [1] is the only well-known trust and reputation testbed of which we are aware. ART has been well supported by the community, and has been used as a competition testbed at a number of conferences.

In ART, agents are art experts, each with varying levels of expertise in different eras. Agents are periodically asked to appraise pieces of art by clients. The accuracy of the appraisals given to clients determines how much business each agent will receive in the future, according to a fixed mechanism used by the testbed. The agent can choose how much to invest in generating its appraisal—greater investment yields greater accuracy. If an agent is asked to evaluate a piece from an era about which he is not knowledgeable, he can seek appraisals from other agents. Agents can also share information with one another about the reliability of other agents' appraisals.

ART is very well-designed for its primary purpose: evaluating agents (who use a number of abilities) in a competitive manner, using a small social trust scenario. ART has a number of desirable properties for a testbed. It offers a well-specified, standardized testing scenario and set of rules. It allows new agents to be easily implemented and plugged into the system; agents can then be used by others for future experimentation. It provides objective metrics for comparison between systems. That said, ART has a number of features that make it less suited for general-purpose trust and reputation experimentation. Other authors (e.g., [9,10]) have noted obstacles to using ART for evaluating their own work.

Under ART, the distinction between buying and selling agents is unclear, making some forms of experimentation problematic. The ultimate purchasers of appraisals (the 'clients') are buyers, and as such, agents serve as sellers for these transactions. Note, however, that clients' method of choosing appraisers (based on past performance) is fixed by the ART specification, precluding experimentation with buyer-side modelling of sellers for these transactions; similarly, it obviates investigation of sellers modelling potentially unreliable buyers. In contrast, as agents buy and sell appraisals with one another, each agent acts as both buyer and seller. Success under these circumstances requires skill in several areas: determining when to make do with your own knowledge, and when to seek help; determining how much to invest in appraisals; determining whom to trust when seeking appraisals; and determining whether or not to be honest when another agent asks you for help. While this is a demanding test, and one appropriate for a competition testbed, it can also obscure the role of each individual skill in an agent's performance. This makes it difficult to isolate individual marketplace components for evaluation. For example, if a researcher wishes to evaluate the performance of a system intended to allow sellers to model untrustworthy buyers, it may not be useful to have the results clouded by the same agent's performance in the unrelated task of deciding whether or not to make honest sales to other agents.

In its role as a competition testbed, ART requires a very well-defined scenario. Unfortunately, this requirement seems to limit the flexibility of the system for experimentation. A number of design choices limit the range of investigations that can be performed using ART. For example, the ART architecture allows decentralized and direct experience models, but precludes testing of centralized models, because the method of sharing information amongst agents is specified by the testbed. It also prevents experimentation with models that regulate an entire marketplace (e.g., mechanism-design based approaches). Features of the chosen scenario prevent investigation of important issues. For example, each appraisal has a fixed price under ART, preventing exploration of vulnerabilities such as Value Imbalance (where a seller builds reputation by honestly executing small-value sales, then uses the reputation gained to cheat on larger ones [5]). The quality of an agent's appraisal is reflected in clients' decisions in the next timestep, preventing exploration of vulnerabilities such as Reputation Lag (where a seller can cheat a large number of sellers for a period of time before his reputation is updated to warn other potential victims [5]).

ART provides a heterogeneous environment where agents share reputation information with agents using other trust and reputation models. To permit communication between agents with different internal models, the format of communication is determined by the ART specification. This imposes a specific trust representation for communication between agents (if not for agents' internal use); the imposed format may not map well to the TRS's native representation, potentially disadvantaging the TRS.

Seeking to clarify why ART is not well-suited for some forms of experimentation, we have focused on a number of its limitations. After so doing, we wish to reiterate that ART is an excellent competition testbed for decentralized social trust and reputation models.

In contrast to the competition focus of ART, the testbed formulation we propose is designed specifically to support general-purpose experimentation and evaluation of trust and reputation technologies. Our proposal is based directly on the platform used in our study of the security of TRS proposals [6]; pertinent aspects of this study are discussed later in the paper.

### 3 Testbed

We sought to formulate a testbed that would support flexible experimentation and meaningful evaluation of trust and reputation technologies. Complete marketplaces may have many TRS components, from a range of possibilities: agents who have individual (and heterogeneous) internal models of other agents' trustworthiness, networks of agents that share reputation information, centralized repositories of reputation data, market-wide mechanisms that regulate trading between agents, etc. A potential adopter of a TRS may have to choose between multiple proposals, despite the fact that the proposals use very different methods internally. An adopter may have to assemble multiple TRS technologies to meet the needs of their complete working system, and may need to understand how

well these components work together. For these (and other) reasons, a testbed will ideally support experimentation with a wide variety of such components. Thus, we set out to design an architecture that was quite flexible.

At the same time, too general a testbed formulation might also be difficult to apply in practical terms. At best, it may be of little benefit to the researcher, leaving much work to be done simply in preparing the testing platform. Worse, a formulation that is too general can make evaluation of TRSes and comparison of results problematic: different researchers are likely to use very different instantiations of the testbed scenario, raising many of the same issues as the author-devised testing that has occurred to date. For this reason, we have specified a well-defined scenario that we believe is useful for a wide range of experimentation. We believe that this is an appropriate and useful balance between flexibility and standardization.

### 3.1 Nature of Tests

For a competition testbed, it is sufficient to supply the testing platform itself; competitors supply the agents, which seek to defeat one another. In contrast, a testbed intended for evaluation and benchmarking requires meaningful tests for candidates to perform. In some fields (for example, computer component benchmarking), a typical approach would be to develop a set of standardized tasks to perform, with well-defined metrics used for comparison. Ideally, the tasks would be representative of real-world demands. For TRSes, however, it is difficult to envision representative ‘tasks’ that do not involve actual interaction with other agents. The most illuminating tests are likely to be those conducted in a realistic scenario, interacting with other agents. Thus, in our formulation, tests consist of two components (in addition to the TRS technology being evaluated): a well-defined marketplace scenario, and a population of agents with which the candidate TRS must cope.

This formulation provides a great deal of flexibility, as well as the ability to test specific components under controlled circumstances. For example, to test TRSes that attempt to allow buyers to cope with cheating sellers, a test would consist of a set of market parameters, and a population of sellers with specific cheating behaviours. These components are experimental controls; each TRS would then be tested against the same scenario, allowing comparison of the results. (This was the approach used in our study [6].) In comparison, to test TRSes that allow sellers to cope with untrustworthy buyers, a test would include a set of buying agents.

Beyond the benefits noted above, this approach has a number of advantages. First, as agents are developed (both TRS technologies, and ‘tests’), they can be made available to other researchers. This allows the test suite to grow, increasing in thoroughness and rigour, as understanding of TRSes increases. (Our set of cheating agents constitutes an initial set of tests, as outlined in a later section.) Second, the standardization of the platform and the availability of agents allows results to be reproduced by other researchers.

### 3.2 Scenario

We sought to develop a testbed that employs a reasonably general scenario, one in which a variety of roles and strategies can be evaluated. For tests to be meaningful, the platform should model as realistic a scenario as practically possible. In the following, the parenthesized parameter values represent settings for a reasonable scenario, one that might constitute a basis for test sets. (These values were used in our own experimentation [6].) While a test specification would include a set of parameter values, the values can be adjusted for experimentation.

We model an ‘advertised-price’ marketplace: sellers offer goods for sale, and buyers choose whether or not to make purchases, and from whom. A fixed set of products (1000) is available for sale. Because we wish to study trust primarily, and not other price-/cost-based forms of competition, the cost to produce/acquire any given good is the same for all sellers. A typical marketplace will have more inexpensive items for sale than expensive ones. To reflect this, the cost of each good is randomly determined using the right half of a Gaussian distribution (i.e., the median occurs at \$0, and probability decreases as price increases). Again, to remove focus from price-based competition, all sellers apply a fixed markup (25% of selling price)—for a given good, all vendors charge the same price.

Each seller is assigned a random number of products that she is able to produce, selected from a uniform distribution (maximum of 10). To reflect the greater availability of less expensive products, the products are again randomly assigned using the right half of a Gaussian distribution (i.e., the median occurs at the least expensive product, with declining probability as price increases).

A simulation run can be populated by an assortment of agents, as desired by the researcher, or as defined in a test specification.

Marketplaces are usually dynamic—traders join and leave regularly. This is important for TRSes, because new agents are unknown (and have no knowledge of other agents), and departing agents result in obsolete knowledge. For efficiency, agents join/exit the market at specific intervals (100 days). After each such interval, each agent departs the marketplace with a fixed probability (0.05). That said, it may be undesirable for the performance of TRSes to be clouded by changes in market size (e.g., profits increasing because the number of buyers increases.) Thus, for every departing agent, one agent of the same type joins, keeping the participant count constant.

### 3.3 Architecture

The testbed architecture is designed to be quite versatile for experimentation, within the constraints of the defined scenario. The architecture is depicted in Fig. 1. In this diagram, BA and SA refer to Buying Account and Selling Account respectively. BE and SE refer to Buying Entity and Selling Entity respectively. All components labelled in boldface italic text are components that are intended to be provided/modified by investigators making use of the testbed. The grey box denotes those components that are observable by marketplace participants,

although this does not imply complete visibility. For example, seller accounts may be visible to buyer accounts, but this does not imply that all seller account data is visible. Such limitations are described in more detail below.

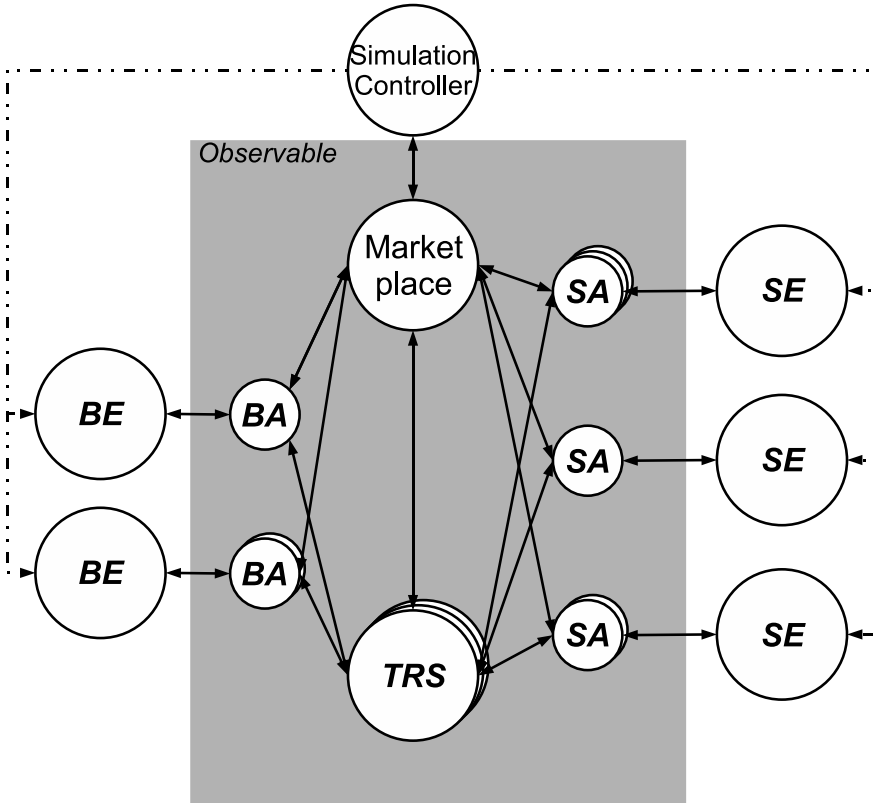


Fig. 1. The Testbed Architecture

A *Simulation Controller* is responsible for actual execution of the simulation. The controller is responsible for triggering each of the day’s events in turn, signaling the appropriate parties when they are required to take action. For example, the controller cues sellers to make product offers at the appropriate times, cues buyers to select products/sellers when offers have been posted, etc.

The scenario makes use of a single centralized marketplace model, represented by a *Marketplace* object. All offers, acceptances, and payments are made through the Marketplace. All accounts reside in the Marketplace, and requests to open accounts are processed through it.

One important aspect of the testbed is the role of TRSes and agents. Some TRSes are implemented entirely centrally, some entirely within the agents; many fall between these two extremes. The use of both agents (represented by accounts

and entities, as described below), and TRS objects, facilitates a wide range of approaches. A *TRS* object implements those components of a TRS that are shared by multiple agents. For example, in a model that makes use of a centralized repository of reputation information, the TRS object would provide that service. TRS objects are useful even in fully decentralized systems, if only to coordinate trust-related actions. For example, when a buyer requests reviews from other buyers, this request would be processed through the TRS, which co-ordinates such communication.

Some important points should be made regarding TRSes. First, as depicted in the diagram, multiple TRSes may be in use simultaneously, for example, by a heterogeneous population of agents. Second, in order to implement a system for experimentation, matching TRS objects and entities typically must be developed. The role of each component is dependent on the characteristics of the TRS in use. For example, in a completely decentralized model, entities may do all reputation tracking and computation; in this case, the TRS might simply serve as the communication channel between agents. At the other extreme, with a completely centralized model, the TRS may perform all reputation-related functions, while entities simply make use of the services provided. This model seems to provide a great deal of flexibility, without undue complication. Third, in some cases (e.g., a market-wide mechanism), a TRS is tightly integrated into the operation of the marketplace itself. For example, Basic Trunits [5] controls what offers may be made by sellers. TRS objects interface with the Marketplace to allow this.

Note that no particular trust representation, or communication protocol is enforced between agents. In fact, it is up to the designer of the TRS component, along with the associated agents, to determine exactly how (or if) communication takes place between agents. This provides support for a wide range of approaches to trust. Note, too, that communication between heterogeneous agents can be supported. Certainly, mapping from one agent's trust representation to another's may be necessary, but the method for doing so is up to the designer of the TRS component used for communication. (This, in turn, also allows experimentation with different means of allowing heterogeneous agents to interact.)

Another important feature of the testbed is the separation of the agent roles into two components: accounts and entities. *Accounts* represent actual user accounts within the marketplace; these are the identities that are observable by other parties in the market. *Entities*, however, represent the actual agents performing actions by using the accounts. Entities are not observable by marketplace participants, reflecting the fact that identity is difficult to establish, particularly in large electronic marketplaces. This distinction is important for a number of reasons. It allows the re-entry phenomenon to be incorporated into experiments (where agents can simply open new user accounts to shed a disreputable identity). It allows for a single agent to control multiple user accounts (as they may in real-world scenarios), as used in attacks demonstrated in our study [6]. It also allows for investigation of collusion. In the case of perfectly loyal and coordinated collusion, a single entity can represent the entire coalition; in the case of



less perfect coalitions, entities can be implemented that communicate with one another outside the observable marketplace. Note that although buying and selling entities are shown as distinct in our architecture diagram, a single entity can play both roles—for example, when controlling both buying and selling accounts to engage in ballot stuffing.

For different components of the marketplace testbed to communicate with one another, certain aspects of communication must be standardized. In our formulation, communication for actual market transactions (i.e., actual purchases) is defined by the specification: the syntax and semantics of product offers, offer acceptances, etc. These are items that are likely to be standardized in a real marketplace situation. Note, however, that the characteristics of communications between TRS components (e.g., exchange of reputation information between agents) are not imposed by the specification, instead left to be determined by those implementing TRS/agents for the system. This ensures maximum flexibility and fair treatment of different TRS approaches, without undermining the standardization of the TRS proposal. Using this architecture, we evaluated five noteworthy TRSes [3,4,5,7,8]. These models use a variety of approaches, including direct experience, witness information, and centralized mechanisms; for each model, agents were able to represent and communicate trust in the native form as described by the authors, without conforming to a specification imposed by the testbed. This demonstrates the versatility of the platform.

Not shown in the diagram is the StatsKeeper module. StatsKeepers are used to accumulate data from runs of the testbed. A default StatsKeeper implementation was used in our experiments, and would be provided as part of this testbed. This module accumulates sales/profit/cheating statistics, by agent group, over time during testbed execution. (An chart illustrating the data accumulated by this module is provided later in the paper.) New StatsKeeper modules can also be easily developed. All objects in the marketplace are visible to StatsKeepers; full data for every sale executed is provided by the Marketplace to the StatsKeeper, so the desired data can be extracted.

Several architectural details are worth noting:

- Buyers do not know of selling accounts until that seller makes an offer. Sellers do not know of the existence of a buying account until it makes itself known by accepting an offer.
- At the time of making an offer, sellers do not know or control whether an offer will be accepted, or by whom.
- A seller can only provide products that she is able to produce. A seller is able to *advertise and sell* (dishonestly) any product, however.
- It is possible for an agent to connect to multiple TRS objects. This allows experimentation with situations where, for example, an agent might make use of shared reputation information with trusted neighbours, as well as accessing data in a centralized repository (i.e., a different TRS).
- Entities can create new accounts at will.

### 3.4 Simulation Execution

Each round represents one day. Each round consists of the following steps (coordinated by the System Controller):

1. After entering into a sale, a buyer will not know whether or not he has been cheated until after some number of days has passed, reflecting processing, shipping, etc; we refer to the rendering of feedback after this *lag* (14 days) as the *completion* of the sale. At the beginning of each day, buyers are notified whether each completing sale was executed honestly or not.
2. After learning about the outcome of each completing transaction, the buyer determines its satisfaction with the transaction (and submits it to the TRS, if using a system that requires immediate reporting.)
3. Upon receiving feedback (if the TRS requires it), the TRS processes incoming feedback.
4. Each buyer's needs are determined for the day. Each buyer is randomly assigned a set of products (up to 5) that it needs to purchase that day; again, these are selected using the right half of a Gaussian distribution, so there is a greater likelihood of needing lower-priced items.
5. Sellers make offers, submitting them to the marketplace. No limits are placed on sellers' capacity or inventory. If a TRS in use regulates market activity, the marketplace consults the TRS for the validity of each order, before posting it.
6. Buyers select the products they wish to purchase, from which sellers, by consulting the posted offers. Buyers are free to consult their TRSes in so doing. For each purchase they decide to make, an offer acceptance is communicated to the corresponding seller account, via the marketplace.
7. Sellers receive the offer acceptances, and have the opportunity to decide if they wish to complete each such sale. Sellers may consult their TRSes to make the decision. For each sale that the seller agrees to make, it decides whether or not to fulfill it honestly or dishonestly. Acceptances are communicated to the marketplace, which forwards each to the corresponding buyer account.
8. Payment is transferred from the buyer account to the seller account, for each sale.
9. Each sale's status (honest or dishonest) is communicated by the seller to the marketplace for storage. This value is not observable by any other marketplace participant, until the buyer is notified during Step 1 of a later round.

### 3.5 Initial Test Set

Much of the value to be gained from an evaluation testbed such as this is the value of the tests: their difficulty, their breadth, and their representativeness of the sorts of issues TRSes might face in a real environment. As an initial set of tests, we provide the set of agents used in our earlier investigations [6]. These agents employ a number of different tactics (consisting only of honest/dishonest transactions that can be executed within the marketplace) based in part on the

problems described in [5]. These agents were designed to test the robustness of TRSes that attempt to cope with dishonest sellers; as such, each agent described below is a seller. They seek to cheat profitably, despite the use of the TRS in question. This test set is far more extensive and difficult than any we have seen used for evaluation of TRSes to date.

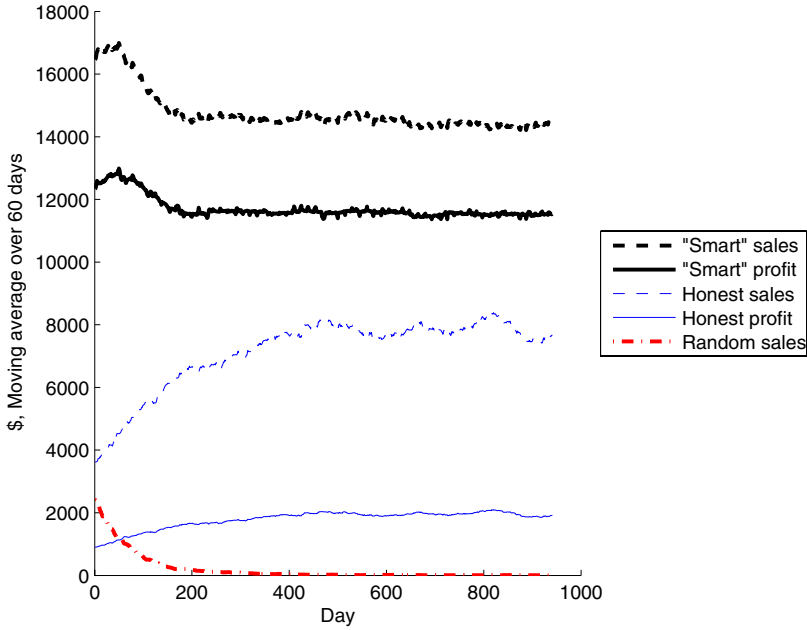
The test set includes the standard, randomly cheating agents employed in so many evaluations. Beyond this, it includes the following agents:

- The **Proliferation agent**, who seeks to win an abnormally large portion of sales by offering products through many user accounts simultaneously, crowding competitors out of the market.
- The **Reputation Lag agent** depends on the delay that exists in many marketplaces between the time he is paid by the buyer, and the time that his reputation is updated to reflect the outcome of the transaction. This agent seeks to build a positive reputation, then use this reputation to cheat as many buyers as possible in the brief period before his reputation is updated to warn other buyers of the change in behaviour.
- The **Re-entry agent**, who creates a new account, attempts to use the non-disreputable status of this new account to cheat as many buyers as possible, then abandons the account and creates a new one, to begin the cycle anew.
- The **Value Imbalance agent**, who seeks to gain good reputation through honest small-value sales, then use that reputation to cheat buyers on large-value sales.
- The **Multi-tactic agent**, who knows how to use all of the tactics described above, and attempts to profitably wield the entire portfolio. This agent is especially intended to undermine the notion of ‘security by obscurity’, that a TRS might be safe from such tactics if the would-be cheater doesn’t know which TRS is in use (and hence its specific vulnerabilities).

As demonstrated in [6], this set of attacks was quite devastating to the set of TRSes evaluated—all of the TRSes were defeated by numerous attacks, and none withstood the Multi-tactic agent.

In actual implementation, we found it useful to decompose our entities into two parts: the actual entity itself, and tactic modules. Each tactic module contains a particular behaviour, for example, the method for launching a particular attack. An entity, then, makes use of one or more tactics to execute its trading activity. This design allows tactics to be re-used. More importantly, it facilitates the design of agents that employ multiple tactics. As noted above, our multi-tactic agents provide an extremely difficult (but realistic and practical) test for TRSes.

Figure 2 depicts one run of the testbed, pitting the Basic Trunits TRS against the Multi-tactic agents. This figure illustrates some of the data generated by the default StatsKeeper module. In this chart, ‘smart’ agents are those employing the multi-tactic approach. Honest sellers, and sellers who cheat randomly are included for comparison. The dashed lines represent the revenue from sales for the day in question (smoothed for presentation), while the solid lines represent profit (i.e., revenue less the cost incurred to furnish the goods). There are an



**Fig. 2.** Basic Trunits, vs. Multi-tactic cheating sellers

**Table 1.** Sales/profit for sellers using multiple attacks

TRS	Cheater sales (% of honest)	Cheater profit (% of honest)
Tran & Cohen	1775.3%	6765.1%
Beta	107.1%	288.3%
TRAVOS	274.6%	613.0%
Yu & Singh	274.9%	723.4%
Basic Trunits	181.8%	577.7%

equal number of agents in each group. Note that the multi-tactic agents are far more profitable than the other groups—cheating is by far the most profitable policy, meaning that the TRS has failed this test.

Table 1 shows the results obtained by the multi-tactic agent against all of the TRSes tested. The first column in each table represents the average sales (in dollars) per multi-tactic cheating agent, relative to those of an honest seller. The second column reflects the profit realized by a cheating agent, relative to an honest one. (Sales are generally more profitable for cheating agents, because they do not incur the cost of honestly furnishing the good.) Results greater than 100% mean that the average multi-tactic agent makes more money than an honest agent. For example, a value of 124% would mean that cheating agents earned 24% more than honest agents per capita. Here, the cheating agents make

far more money than honest ones—quite troubling, because this means that a profit-maximizing agent should choose to cheat rather than be honest. All of the TRSes have failed the test.

This set of tests is certainly not exhaustive. Further, it only tests TRSes against dishonest sellers, rather than dishonest buyers, agents that lie to one another about their opinions of other agents, etc. Nonetheless, it constitutes a substantial initial test suite for this important aspect of TRS operation.

## 4 Conclusions and Future Work

The trust and reputation testbed described in this paper allows a breadth of experimentation and a thoroughness and objectivity of evaluation that have previously been unavailable from publicly-available, standardized testing tools. The design of this testbed is a proven one, having been used to shed light on important issues that had previously been unexplored experimentally—in particular, the degree to which existing TRS proposals can withstand cheating agents that actively attempt to circumvent the protections of the system. The platform has shown itself to be flexible, supporting experimentation with TRSes using a variety of approaches.

With this design, we have attempted to allow the greatest degree of flexibility of experimentation possible, while still providing ease-of-use and the ability to generate meaningful benchmarks and comparisons. As a result, not every TRS can be tested using this platform: for example, ones that do not function in marketplace environments.

The testbed is intended to allow more thorough testing than has typically been performed for TRSes in the past. As TRS researchers develop new agents, the test suite will grow, increasing the rigour of the evaluations, and the insights provided.

While we believe this testbed to be an important tool in itself, and a significant step towards improving the evaluation of TRSes, we do not contend that it is perfect or complete in its current formulation. We hope that this proposal serves as a basis for discussion and development within the trust and reputation community. The value of this tool will increase with input from other researchers, ensuring that the platform is complete, and flexible enough (within the limits of practicality) to meet their needs. Future work includes consultation with researchers to identify potential refinements and achieve consensus regarding:

- Scenario;
- Architecture;
- Recommended parameter values for benchmarking;
- Range of support for agents, and trust and reputation technologies.

Through these consultations, we will be diligent to ensure an effective balance between the complexity that might be introduced with increased flexibility, and the standardization that facilitates benchmarking and usability.

## References

1. Fullam, K.K., Klos, T.B., Muller, G., Sabater, J., Schlosser, A., Topol, Z., Barber, K.S., Rosenschein, J.S., Vercouter, L., Voss, M.: A specification of the Agent Reputation and Trust (ART) testbed: experimentation and competition for trust in agent societies. In: *AAMAS 2005: Proceedings of the fourth international joint conference on Autonomous Agents and Multiagent Systems*, pp. 512–518. ACM, New York (2005)
2. Teacy, W.T.L., Huynh, T.D., Dash, R.K., Jennings, N.R., Luck, M., Patel, J.: The ART of IAM: The winning strategy for the 2006 competition. In: *Proceedings of the Sixth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2007) Workshop on Trust in Agent Societies*, Honolulu, Hawaii, USA (2007)
3. Teacy, W.T., Patel, J., Jennings, N.R., Luck, M.: Travos: Trust and reputation in the context of inaccurate information sources. *Autonomous Agents and Multi-Agent Systems* 12(2), 183–198 (2006)
4. Yu, B., Singh, M.P.: Distributed reputation management for electronic commerce. *Computational Intelligence* 18(4), 535–549 (2002)
5. Kerr, R., Cohen, R.: Modeling trust using transactional, numerical units. In: *PST 2006: Proceedings of the Conference on Privacy, Security and Trust*, Markham, Canada (2006)
6. Kerr, R., Cohen, R.: Smart cheaters do prosper: Defeating trust and reputation systems. In: *Proceedings of the Eighth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2009)*, Budapest, Hungary (2009)
7. Jøsang, A., Ismail, R.: The beta reputation system. In: *Proceedings of the 15th Bled Electronic Commerce Conference e-Reality: Constructing the e-Economy* (June 2002)
8. Tran, T., Cohen, R.: Improving user satisfaction in agent-based electronic marketplaces by reputation modelling and adjustable product quality. In: *AAMAS 2004: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, Washington, DC, USA, pp. 828–835. IEEE Computer Society, Los Alamitos (2005)
9. Hang, C.W., Wang, Y., Singh, M.P.: An adaptive probabilistic trust model and its evaluation. In: *AAMAS 2008: Proceedings of the 7th international joint conference on Autonomous Agents and Multiagent Systems*, Richland, SC, International Foundation for Autonomous Agents and Multiagent Systems, pp. 1485–1488 (2008)
10. Harbers, M., Verbrugge, R., Sierra, C., Debenham, J.: The examination of an information-based approach to trust. In: Noriega, P., Padget, J. (eds.) *International Workshop on Coordination, Organization, Institutions and Norms (COIN)*, pp. 101–112. Durham University, Durham (2008)

# Evaluating the STORE Reputation System in Multi-Agent Simulations

Jonas Andrulis<sup>1</sup>, Jochen Haller<sup>1</sup>, Christof Weinhardt<sup>2</sup>, and Yuecel Karabulut<sup>3</sup>

<sup>1</sup> SAP Research, Vincenz-Priessnitz-Str. 1, 76131 Karlsruhe, Germany  
`firstname.lastname@sap.com`

<sup>2</sup> Information Systems and Management, University of Karlsruhe,  
Englerstr. 14, 76131 Karlsruhe, Germany  
`firstname.lastname@iw.uni-karlsruhe.de`

<sup>3</sup> SAP Labs, USA  
`firstname.lastname@sap.com`

**Abstract.** In recent global business environments, collaborations among organisations raise an increased demand for swift establishment. Such collaborations are formed between organisations entering Virtual Organizations (VOs), crossing geographic borders and frequently without prior experience of the other partner's previous performance. In VOs, every participant risks engaging with partners who may exhibit unexpected fraudulent or otherwise untrusted behaviour. In order to cope with this risk, the STochastic REputation system (STORE) was designed to provide swift, automated decision support for selecting partner organisations in the early stages of the VO's formation. The contribution of this paper first consists of a multi-agent simulation framework design and implementation to evaluate the STORE reputation system. This framework is able to simulate dynamic agent behaviour, agents hereby representing organisations, and to capture the business context of different VO application scenarios. A configuration of agent classes is a powerful tool to obtain not only well or badly performing agents for simulation scenarios, but also agents which are specialized in particular VO application domains or even malicious agents, attacking the VO community. The second contribution comprises of STORE's evaluation in two simulation scenarios, set in the VO application domains of Collaborative Engineering and Ad-hoc Service provisioning. Besides the ability to clearly distinguish between agents of different classes according to their reputation, the results prove STORE's ability to take an agent's dynamic behaviour into account. The simulation results show, that STORE solves the difficult task of selecting the most trustworthy partner for a particular VO application domain from a set of honest agents that are specialized in a wide spread of VO application domains.

## 1 Introduction

Virtual Organizations (VOs) have emerged as a business model in application domains with a high demand for cross-domain collaborative business processing. Increased collaboration among business partners and focusing on an organization's

core competencies requires such novel models to address business opportunities. A VO is defined as a set of sovereign, geographically dispersed organizations that temporarily pool their resources to jointly address a business opportunity one organization alone is not able to master [1]. A VO follows a phased life cycle where speed is an essential requirement. Especially the initial identification and formation phases, dealing with potential VO partner identification, selection and integration have to be swiftly conducted to stay ahead of competitors. Afterwards, the VO enters the operational phase, executing Business Processes, while the evolution phase is entered whenever exceptions occur or compensation is necessary. The dissolution phase concludes the VO life cycle, dispersing the VO assets among the participants. VOs form for instance in response to a government issued tender in the Collaborative Engineering (CE) application domain, e.g. to upgrade a plane or in domains such as Ad-Hoc Service Provisioning (AH), e.g. offering electronic services to business travellers. The VO manager role (the trustor) is responsible for the decision making which potential members (the trustees) are invited and subsequently selected to join the VO, a crucial decision with respect to the entire VO's success [2]. The possibility of a VO partner performing badly during the VO's operational phase or announcing bankruptcy endangers the investment taken in integrating their processes and infrastructure for the purpose of the VO. A reputation system can provide additional decision support besides the a priori knowledge from quotations and bidding to avoid events such as VO partner replacement by minimizing the risk [3] of choosing unreliable partners in the first place. The STochastic REputation (STORE) system for VOs [4] is designed to offer exactly this kind of decision support. It aggregates observable data about a trustee's behavior, following principles of probability theory. To achieve this, reputation, an objective trust measure, is aggregated from multiple independent trust sources, so-called Trust Indicators (TIs), inherently characterizing an organization's reliability. Such TIs provide measurable trust values about an organization from heterogeneous domains, e.g. capturing operational aspects such as timely delivery of a service, organizational stability measuring employee turnover or environmental risks due to factory locations. Trust aspects captured by the TIs are, in contrast to reputation, highly subjective properties conveying the probability of an organization's expected reliable behavior [5]. To allow for the desired predictions of an organization's future performance, a stochastic modeling approach is chosen [6]. More concretely, this approach integrates a stochastic trust management approach in a reputation system.

This publication provides a simulation based evaluation of the STORE reputation system. The contribution is three-fold:

1. Design and implementation of a Multi-Agent based Simulation (MAS) framework to evaluate the STORE reputation system.
2. An interface extension of the STORE reputation system offering a trustor the capability to express application scenario specific trust preferences about a trustee.



3. Evaluation of reputation based VO member selection in two different VO application scenarios from Collaborative Engineering (CE) and Ad-Hoc (AH) service provisioning.

In the following section, related work from Trust and Reputation Management as well as MAS is discussed. Section three starts off with a brief introduction of STORE's model and architecture, followed by the detailed design and implementation of our simulation framework. Section four then comprises of the evaluation results from the two VO application scenarios. Section five concludes and offers an outlook on future work.

## 2 Related Work

This section provides a brief enumeration of related work from the research field of Trust and Reputation Management, followed by a more detailed assessment of related work in the area of (Agent-Based) Simulation.

Trust is a complex sociological phenomenon. In the context of a VO, we define trust as the subjective probability by which the trustor expects the trustee to perform actions captured in a role specification within the context of a VO. This definition is based on work from Gambetta [5] and Jøsang [7]. In the area of information technology, the term trust management was introduced by [8], who defined the term "trust management (problem)" as the collective study of security policies, security credentials and trust relationships. This technical perspective resulted in a system providing access control for distributed environments. Following this ground breaking publication, a multitude of trust management approaches were developed and published in parallel. The most recent and successful ones are surveyed in [7]. On the higher level of business to consumer e-commerce, Egger et al. [9] present a related trust model, directly observable factors that characterizing online vendors. While the general approach to root trust in observable indicators is comparable to our work, the application domain and hence the relevant indicators are different.

Reputation is a known concept in many disciplines, equally broad as and closely related to trust [10]. Reputation is frequently interpreted as the general opinion of a group towards an individual, another group of people or an organization. Broken down to the field of trust management in VOs, we define reputation as the perception a VO has about the intentions and norms of another organization. This perception develops through past actions and through objective indicators. A general reputation can thereby be mapped to an individual binary (directed) trust relationship. Related reputation systems based on stochastic principles are Jøsang's Beta Reputation System [11], TRAVOS [12] and Regan's Bayesian modeling approach [13]. While the Beta and TRAVOS reputation systems are based on feedback from personal experience, the latter, in the same manner, relies on modeling with one single distribution. In contrast, STORE takes a different approach with a richer trust model, rooting trust in a set of trust sources instead of one. Each TI is modeled with a TI specific probability distribution function instead of the same. This approach then demands

a more sophisticated aggregation of the trust sources. While for instance the Beta distribution itself aggregates the homogeneous trust sources to a reputation value, STORE's heterogeneous trust sources require an aggregation layer based on a tree shaped Bayes Network. The Beta Reputation System was recently extended to multi-valued feedback with contributions from one of the authors [14].

Reputation systems, mostly focusing on systems using transaction feedback as only input parameter, are frequently evaluated using MAS. Two reputation systems, following a probabilistic approach comparable to the STORE system, are the TRAVOS model by Teacy et al [12] and the Beta reputation system by A. Jøsang et al. [11]. To evaluate these systems, a MAS marketplace was created. We also adopt a similar agent based framework design, where different kinds of agents interact. This approach is able to capture a high degree of complexity, as encountered in VO environments. In many MAS models, "bad behavior" is a choice an agent takes out of self-interest. This is not only the case for the two examples mentioned above, but also for the large class of reputation systems that are evaluated by playing a prisoner's dilemma game. Lik Mui et al. [15] summarize the recent work in this area and illustrate the resulting strategies. In these models the real quality of the agent is determined by his choice of strategy in a game, that allows for exactly two different strategies and the reputation is an indicator for the kind of strategy a certain agent is playing. Our approach differs in that our agents do not "choose" to perform badly, but cooperate with their best possible quality, while not every agent is capable of offering the same high quality. We believe, this approach delivers a better model for business environments, where the outcome for all participants heavily depends on the quality of the individual inputs.

All related reputation systems define one function to compute a reputation value. By offering, in addition to a consolidated reputation value, reputation values for four trust classes, STORE offers several functions that allow the setting of application specific trust preferences per class. Furthermore the change of an agent's behavior is swiftly captured as an adjusted reputation value.

### 3 Simulation Framework

This section introduces the MAS framework, designed to evaluate the STORE reputation system. First, the requirements derived from both VO scenarios are analyzed, accompanied by a brief scenario description. Second, the agent model and framework design is presented. Detailed information about the framework's setting for the different scenarios conclude this section.

#### 3.1 Requirements from Application Scenarios

The goal of the STORE reputation system design, as published in [4], is to provide reputation based decision support for selecting trustworthy partners in VOs. Such support is intended to be delivered for all classes of VOs and not

a specific class in particular. The EU funded IST project TrustCom published, as one of its deliverables, a VO classification along with class specific security and trust requirements [16]. The outcome of this classification exercise was a nearly continuous set of VO classes that could be distinguished, among others, according to properties such as expected lifetime and expected volume of business transactions. In this publication, we will evaluate STORE in the two contrasting VO scenarios from this classification.

This is, first, a CE VO from the aerospace industry, addressing a government issued tender for a plane upgrade. A VO manager such as British Aerospace Systems, enacting the manager role, seeks to attract the most trustworthy design specialists, several storage providers and high performance computing providers. In summary, the the integrator seeks around five trustworthy organisations to start a VO. The operation then starts with the integrator submitting design upgrade tasks to the design specialists. The data is maintained by a storage provider and retrieved for design activities from this location by the design specialists. Updated design data is uploaded back to the storage provider. A similar work pattern then continues with the design validation where the design specialist simulates the design data on the high performance computing provider's infrastructure and e.g. reports faulty aerodynamic behaviour, triggering another design step. This VO requires weeks to be set, may last up to several decades and single transactions conducted among the members reach a high financial volume, up to the M€.

In an AH VO, the members, consisting of mobile operators, local and global electronic service and infrastructure providers as well as restaurants and hotels, provide electronic services to business travellers. Assuming the VO manager maintains for instance the local WLAN network infrastructure, at least two organisation - the traveller's home operator and a content provider - are needed for a VO. Such a traveller, residing at a foreign location e.g. in a hotel (enacting the VO manager role) seeks access to electronic services providing data and other digital content. Such services range from local tourist information, weather forecast or entertainment to the traveller's home operator's portal with e-mail or stock information. A telco scenario, consisting of the hotel as integrator, local and home operator as well as the service providers is set up in seconds. It may last for hours up to days (billing in the end taking most of the time) and transactions with little volume are conducted. For the remainder of this paper, these scenarios will be abbreviated as the aerospace and telco scenarios.

To judge the claim of STORE being able to provide reputation based decision support for VO scenarios in general, the evaluation methodology must allow a comparison of the reputation's decision support on the partner selection across different scenarios. Due to STORE's Bayesian approach on reputation computation, as described in the following sections, and the intrinsic uncertainty of an organisation's trustworthy behaviour, the reputation mechanism can not be analysed analytically in closed form. We adopted a MAS methodology because it is, to our knowledge, the only choice that integrates the stated VO properties as simulation settings and meets all other requirements and conditions.

### 3.2 The STORE Reputation System

The left box in Figure 1 provides an overview of STORE's model and architecture, as far as required to follow the coming sections in this paper. The system's details are published in [4]. Organizations are represented by their TIs, characterising their observable trustworthy behaviour. "Delivery Delay" is a sample TI, characterising one operational aspect of the organization's trustworthy behavior. TIs are classified into  $m = 4$  Trust Classes (TCs) of environmental, financial, operational and organizational TIs. This taxonomy of TCs is extensible, further classes can be defined. The TCs are derived from extensive research in domains related to reputation management such as Operations Research, Risk Management [4]. Each TI is modelled with a Probability Distribution Function (PDF), which is updated with each new data observation, according to TI specific update intervals. TIs are aggregated towards the reputation vector using a Bayes Network<sup>1</sup>. TIs are modeled in the leaf nodes. Their heterogeneous PDFs are normalized according to a common ordinal scale by the set of preference or  $\pi$ -nodes in the network. With the help of the preference nodes, the heterogeneous TIs become comparable. All TIs are aggregated by the generalized reputation node  $R$ , while the TC nodes only aggregate the TIs belonging to their class. The STORE reputation system finally delivers a reputation vector consisting of the TCs and  $R$  to a reputation requester. Each vector component's value consists of the expectation value of its node's PDF and is defined in the interval  $[0, 1]$ . This reputation vector extends the originally in [4] published interface which only returned the generalized reputation value  $R$ . The new TC components allow a VO manager to express trust preferences according to his VO application context, e.g. by emphasising operational trust aspects. The form, trust preferences are expressed in, is described in Subsection 3.3.

### 3.3 Agent Model and MAS Framework

The MAS framework simulates a full VO lifecycle, one in each round. Agents, representing organisations, are matched to form a VO based on their reputation in each round. An agent enacts exactly one of the two VO business roles, manager or member. Its behavior is determined by a set of modelled TIs which characterize this agent's observable, trustworthy behaviour. In this paper, we focus on the six TIs, listed in Table 1, covering all four TCs. TIs are observed in regular time intervals. From these TI data observations, an agent specific reputation vector is computed by STORE and the MAS framework derives a QoS value (defining the agent's production input) in the interval  $[0, 1]$  from a scenario specific QoS rule set. A value of one is interpreted as the maximally achievable performance. The framework also maintains a virtual "bank account" for each agent. The right hand box in Figure 1 illustrates these described components and their relations.

<sup>1</sup> A Bayes network is graphically represented as a directed, acyclic graph; nodes maintain random variables, edges denote conditional probability relationships between the connected nodes.

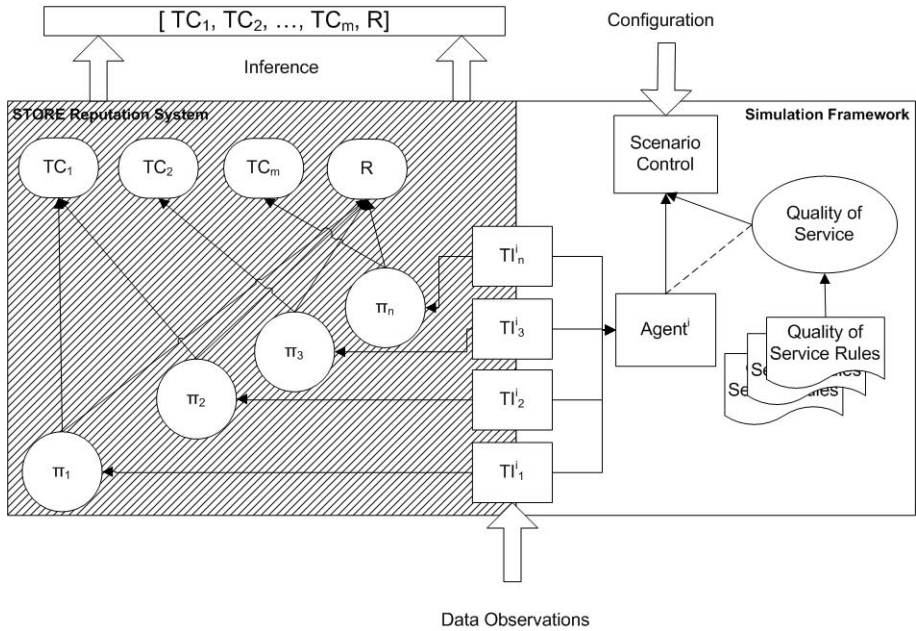


Fig. 1. STORE and MAS Framework architecture

Agent classes are assigned to agents, configuring their behaviour for a planned simulation. For this paper, four agent classes are defined. Class 1 defines a uniformly, in all TCs, well performing agent, Class 2 defines an agent who is specialized in telco scenarios and performs well in the operational and organizational TCs. Similarly, Class 3 defines an aerospace scenario specialized agent which performs especially well in the financial and environmental TCs, while Class 4 defines a uniformly badly performing agent. Table 1 lists the different agent classes and as well as their performance with respect to TIs and TCs.

The framework generates TI data for each agent by drawing from each individual TI's inverse PDF. Technically, assigning a class to an agent sets the inverse

Table 1. TI settings for different agent classes (+ high, - low)

TI	Trust Class	Class 1	Class 2	Class 3	Class 4
Country Bond Spread	Environmental	+	-	+	-
Cash Flow Margin	Financial	+	-	+	-
Complaint Rate	Operational	+	+	-	-
Delivery Delay	Operational	+	+	-	-
System Downtime	Operational	+	+	-	-
Employee Fluctuations	Organizational	+	+	+	-

PDF's parameters, hereby determining the agent's later behaviour during the simulation. For example the Delivery Delay TI follows an Exponential distribution with one parameter  $\lambda$  and its mean  $\frac{1}{\lambda}$ . Assigning Class 4 to an agent sets  $\lambda$  to a smaller value, hereby shifting the exponential distribution's entire support and its mean towards higher values of delay. For instance setting  $\lambda = \frac{1}{2}$  results in an expected delay of 2 (days). Assigning Class 1 with e.g.  $\lambda = 1$  would result in the opposite, a shift towards smaller delay times of one (day). Adopting this statistical approach allows for a realistic model of an organization's trustworthy behaviour. Business partners typically follow a certain strategy, e.g. honest or malicious, which is captured by the agent classes and the inverse PDF's mean. Furthermore, organisations show small deviations around this strategy in their behaviour. Even honest ones may exhibit e.g. temporary lapses in trustworthiness due to events out of their control such as accidents or infrastructure malfunction. This volatility is captured by setting the inverse PDF's variance.

The simulation differentiates agents into managers and members where an agent of a certain type can only interact with his counterpart. Agents are matched with the Gale and Shapley [17] algorithm. Each agent computes a preference relation over every agent of the opposite VO role that is only based on the agent's reputation vector. Such a matching leaves no incentive for any two partners to enforce a different matching and is therefore called "stable". Adding now the VO application domain context to the matching, the reputation vector  $\vec{R} \in \mathbb{R}^5$  for agent  $i$  as provided by STORE consists of a value for each TC ("environmental", "financial", "operational" and "organizational"), as well as the generalized reputation value  $R$ .  $\psi$  defines the metric for an agent-specific preference relation,  $\psi : \mathbb{R}^5 \rightarrow \mathbb{R}$ , as  $\psi(\vec{\omega}) = \vec{R} \cdot \vec{\omega}$  where  $\vec{\omega}$  captures the agent specific "weighting" that sets the trust preferences for  $\omega$ 's components. For example  $\vec{\omega}_0^T = (0\ 0\ 0\ 0\ 1)$  represents a preference vector that only incorporates the generalized reputation value and ignores the trust classes. A scenario specific preference vector for the separate TCs captures their relevance in different VO scenarios.

The MAS framework entails a scenario control component, controlling the progress of a running simulation scenario. Simulations are executed in rounds, in each round performing the following actions:

- Generate TI data for every agent.
- Calculate a "Quality of Service" (QoS) value for every agent from the TI data according to the QoS rule set.
- Request the updated reputation vectors from STORE for every agent.
- Compute a preference relation for each agent over all potential partners of the opposite VO role.
- Perform matching according to the preference relations. The matched agents form a new VO.
- Calculate production for newly formed VO and resulting pay-off for every agent.
- STORE receives the TI data as new data observations, resulting in a Bayes Network update.

The production or productivity  $y$  of a VO is bilaterally calculated using a Cobb-Douglas productivity function  $y = x_{Manager} \cdot x_{Member}$ , where in our model the input factors  $x_i$  are the QoS values. They are dimensionless and represent a productivity rate compared to the possible optimum. The productivity between a manager and member is equally divided between both parties, resulting in a pay-off rate for every agent. A higher QoS of the transaction partners leads to an increased pay-off for both of them. This way, we create an incentive for each agent to transact with a partner of higher QoS and a method to assess how well STORE compute's an agent's reputation by comparing it with the cash rates of agent representatives from the four agent classes.

The MAS framework is tailored for STORE's evaluation. This becomes apparent when examining the TIs characterising an organisation's trustworthy behaviour. The MAS takes the aggregation of multiple, heterogeneous trust values towards a reputation value into account. The framework can very well be used for the evaluation of other reputation systems, e.g. aggregating homogeneous feedback values, that provide a less challenging evaluation task. The QoS rules serve well as a measure to compare different reputation systems with each other in the same application scenario.

### 3.4 Technical Scenario Settings

The MAS framework is implemented in the Java programming language and offers a set of configuration options to define VO application domain specific settings. First, the agent set up can be freely defined including the number of manager and member agents, as well as their class assignments. The agent ratio, how many members are required to form a VO with one manager is separately defined. Regarding the general simulation properties, the number of rounds is part of the setting, along with a number of so-called blind rounds. The latter do not contribute to the generated result sets, but are defined to bridge the system's tune in phase. This allows for equal, well defined cold start conditions of simulations which are supposed to be compared afterwards. To avoid large unjustified financial increases of the agent's accounts, transaction costs per round can also be set. The length of one round in real time and the number of total rounds played, vary in both selected scenarios. The length of a round is an important factor for the reaction speed of the reputation value and should be comparable to the expected lifetime of the VO (when simulating a VO with an expected lifetime of 10 years with rounds that last for one week real time does not make sense).

## 4 Evaluation

In this section, the results of our evaluation are presented. First, the setting for the two VO scenarios is described. Second, we show how STORE is able to separate agents of different classes by reputation. Third, as an additional reputation independent measure, an agent's cash rate is introduced and used to

compare simulation results across different VO application scenarios. A reputation system independent random matching for the potential partners is used as a benchmark against the two scenario specific STORE matchings. The same simulation results are used to discuss a VO's productivity increase when relying on STORE's decision support. The Section concludes with a sensitivity analysis. All presented data is average over 50 runs of the identical scenario configuration. A transaction cost per round and agent of 0.15 currency units is set.

#### 4.1 Simulation Setting

The aerospace environment from [3.1](#) consists of few organisations with few high-turnover transactions, due to the high initial investment necessary for joining. The aerospace market in our simulation consists of one Class 1 manager seeking five member agents for the formation of an aerospace VO. There are in total 9 potential members: 2 Class 1, 2 Class 2, 2 Class 3 and 3 Class 4 agents. One round lasts for about half a year real time and 20 rounds are played in one scenario. The STORE system is configured to react slow to change. The CE weighting vector  $\vec{\omega}_{CE}^T = (0.5 \ 0.5 \ 0 \ 0 \ 1)$  includes, with weight 0.5, the TCs "environmental" and "financial" along with the generalized reputation value. The two trust classes are emphasised, since physical goods are transported in such VOs relying on proper infrastructure and since high volume transactions take place. This preference vector takes, by preferring the less frequently observed "environmental" TC, also the long term orientation of an aerospace scenario into account. These TCs tend to capture a sustainable development with the rarely observed TIs "Country Bond Spread", capturing a region's environmental risk, and "Cash Flow Quote".

In the telco scenario from [3.1](#), many transactions are executed during one day and the typical transaction volume is low. This lowers the entry barrier for new and small companies. In the telco simulation environment, 3 managers (2 Class 1 and 1 Class 4) are matched with 2 members each. On the market there are 1 Class 1, 2 Class 2, 2 Class 3 and 2 Class 4 potential members available. The STORE reputation system reacts quickly to change. 50 rounds are played where one round represents about five days real time. The telco trust preference vector  $\vec{\omega}_{AH}^T = (0 \ 0 \ 1 \ 0 \ 1)$  only weights the TC "operational", as this class focusses on the current quality of service ignoring long-term trends, and again the generalized reputation value. In such a short-lived environment, fast and timely service ranging from low-level network speed to higher-level content delivery of stock data is essential. For example the TIs "System Downtime" and "Delivery Delay" belong to this TC and are both frequently observed. In this setting, the sums of both trust preference vector components are equal, simplifying cross-VO scenario comparisons.

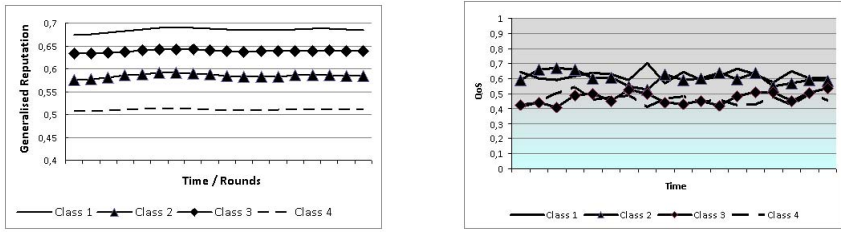
#### 4.2 Separation of Agents from Different Classes

To illustrate how STORE can separate agents by reputation according to their class assignments, we take a closer look at the development of the reputation values compared to the reputation independent QoS measure. This first simulation



evaluates this very basic function in a static setting without dynamic changes in the agent's behavior.

Figure 2 depicts the reputation of the four classes of agents over the simulation rounds. The generalized reputation value of the two specialized agents (Class 2 and 3) is clearly, as desired, in between the reputation of agents from classes Class 1 and 4. This good result becomes even more appreciated when compared to the QoS measurements in Figure 3. While the agents representing the four classes are clearly and continually separated by reputation, their actual, real performance is not. For instance the measured QoS for the agents of Classes 1, 2 and 3 even intersects in round three. The agent classes are on purpose defined that the resulting inverse PDF parametrisations constitute a worst case where agent classes only marginally differ in their real performance measured by QoS. This first functional test scenario was conducted in a telco scenario setting without introducing a VO context in form of a trust preference vector yet.



**Fig. 2.** Generalized Reputation of the **Fig. 3.** QoS of the four agent classes four agent classes changing over time in changing over time in an telco scenario an telco scenario

This neglect of VO context and TCs explains, why the Class 3 agent carries a higher generalized reputation value in a telco scenario setting.

The following simulation scenario now addresses this lack of VO context by joining results from both, telco and aerospace scenario simulation runs, compared with the random matching benchmark scenario. Each scenario is executed with its specific setting and the manager agents use the above defined trust preference vectors  $\vec{w}_{CE}$  and  $\vec{w}_{AH}$ . If STORE delivers the desired, VO specific reputation based decision support, the most trustworthy agents (with the best QoS for their VO scenario) should interact with each other and thus receive better pay-off rates. Their overall cash rate should be higher than in the random matching benchmark. To cross-evaluate the newly extended reputation interface with the trust preference vectors, we additionally evaluate each scenario with the correct scenario specific trust preference vector against the same setting with the vector from the other scenario (an aerospace scenario with weighting vector  $\vec{w}_{AH}$  and vice versa). We expect, that with the correct scenario specific trust preference vectors, agents specialized in the same VO domain are preferably selected along with Class 1 agents, resulting in higher cash rates. The random matching

benchmark represents a VO environment with no reputation based decision support.

The resulting cash rates for these three configurations are illustrated in the following Figures, first clearly indicating the superior performance of the reputation based matching compared to the random matching. In the random matching benchmark (Figure 4), every agent performs badly since being randomly matched even to Class 4 agents. In all scenarios and with both trust preference vectors, the Class 1 and 4 agents are continually clearly separated by their cash rates and, as designed and expected, exhibit superior and inferior performance respectively. Figure 5 shows the results of an aerospace scenario with AH trust preferences, as e.g. expressed by an uninformed VO manager. As shown in Section 4.2, the wrong trust preferences still manage to separate Class 1 and 4 agents, but struggle to identify scenario specialized agents. The specialized Class 2 and 3 agents perform better as in the random matching, but are not separated as well as with the correct CE trust preferences depicted in Figure 6. This scenario, assuming an informed VO manager, delivers the best cash rates, by preferring Class 1 and the CE specialized Class 3 agents at all times (overlaid lines of both classes in the diagram). The Class 2 agent still receives a pay-off comparable to the random matching benchmark and the Class 4 agent is only selected if absolutely necessary.

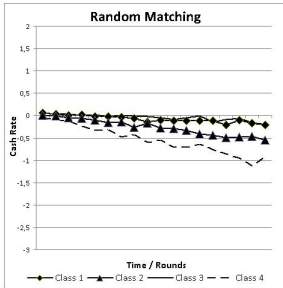


Fig. 4. Cash rates for the four agent classes in a random matching aerospace scenario

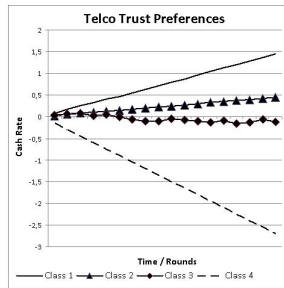


Fig. 5. Cash rates for the four agent classes in an aerospace scenario with wrong trust preferences

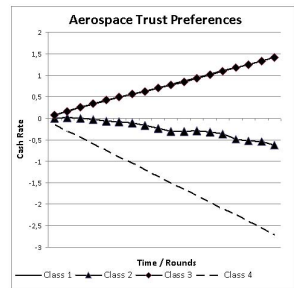
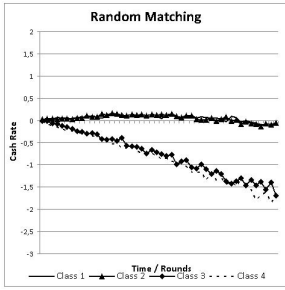


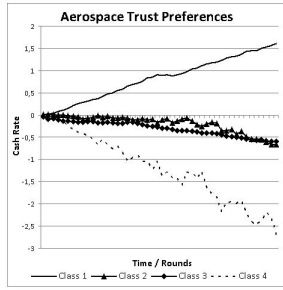
Fig. 6. Cash rates for the four agent classes in an aerospace scenario with correct trust preferences

This clearly shows that STORE’s reputation based decision support for selecting members of VOs in general meets the expectations, but at the same time emphasizes the importance of VO scenario specific trust preferences expressed by an informed VO manager. The reputation based matching executed with an improper trust preference vector generates significantly worse results especially for the specialized agents. The trust preference vector is an exogenous input in this simulation and has to be set by a business expert, the VO manager.

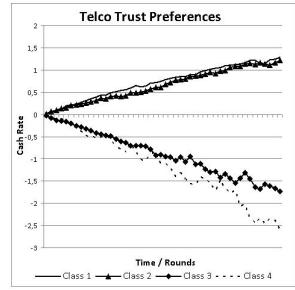
The results in an telco scenario are similar to the ones presented above: The reputation based member selection, using a wrong trust preference vector



**Fig. 7.** Cash rates for the four agent classes in a random matching telco scenario



**Fig. 8.** Cash rates for the four agent classes in a telco scenario with wrong trust preferences



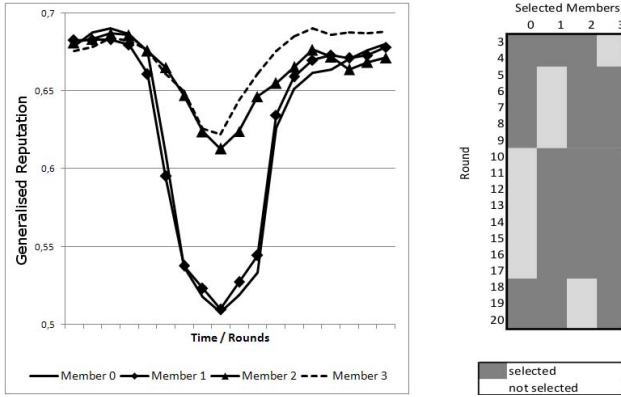
**Fig. 9.** Cash rates for the four agent classes in a telco scenario with correct trust preferences

(Figure 8) manages to separate Class 1 from 4 agents, but only a correct trust preference vector (Figure 9) is capable of differentiating specialized agents.

### 4.3 Sensitivity Analysis of Dynamic Agent Behavior

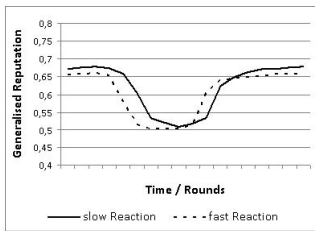
Many reputation systems struggle when observing the effects of a change in an agent's behavior. In our MAS framework, an agent's behaviour can be changed in each round with well defined configuration settings. With this powerful tool, research with more than the four previously defined, static agent classes becomes possible. We can simulate VO environments with agents exhibiting decaying or increasing performance, even modelling attacker agents with, for instance, oscillating performance, becomes possible. This allows to evaluate the STORE reputation system with different configurations, for instance reacting faster or slower to changes in an agent's behaviour. We begin with a scenario of four equally defined member agents. We configure one manager seeking three members to form a VO, so that every round, one of the potential members is not selected for the VO. Due to STORE's stochastic model, the observed generalized reputation value for the four agents is initially at a similar level with random diffusion. Which one of potential members is selected by the manager is random in these first rounds. After some rounds we change the behavior of all agents: two of the potential members (0 and 1) suffer a large decrease in their performance, while the other two (2 and 3) only decrease slightly. In Figure 10, left hand chart, we observe a significant change in the generalized reputation value for the first and a less severe for the second group. This affects the selection behavior of the manager which now prefers the members 2 and 3 for the rounds five to 17, suffering from decreased performance, in the right hand chart. When all agents recover to their initial performance, the selection becomes random over the entire group of potential members again.

This illustrates that STORE reacts well to gradual change in an observed agent's behavior and that it enables every manager to react to an altered situation, by supporting his decision to select only the most trustworthy partners at all times.

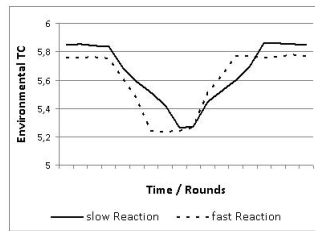


**Fig. 10.** The generalized reputation values for four potential members changing their quality over time and a table indicating which agents are selected for VO membership

The following scenario evaluates STORE’s reaction speed to changes in an agent’s behavior. This speed heavily depends on a STORE internal configuration, how far back in time observed TI data is considered for the reputation computation. A larger time window results in slower reaction which is suitable for a long-lived aerospace scenario, while a smaller time window and quicker reaction is more appropriate for an agile telco scenario. An exemplary comparison of the reaction speed for both configurations is shown for the generalized reputation value change in Figure 11 and for the "environmental" TC change in Figure 12. This same change of behavior, occurring in round four, as in the previous setting is employed.



**Fig. 11.** Reputation reaction to quality change with STORE configured for slow and fast reaction



**Fig. 12.** TC reaction to quality change with STORE configured for slow and fast reaction

We observe that STORE can be adapted to react slower or faster to behavioral changes, as demanded by a particular VO scenario. In fast paced telco scenarios, a change in the computed reputation can already be observed within the same round and fully adapts to the change within two more. A slower configuration, better suited for long lived aerospace scenarios, requires two more rounds which is still acceptable since one round amounts for appr. half a year, while aerospace

scenarios may last for decades. However, configuring STORE's reaction speed should be carefully planned, since extreme settings in both directions, e.g. allows attacks on the reputation mechanism. While reacting too fast, older TI data is disregarded and an attacker may aim at boosting his reputation with well performed, low volume transactions, while reaping the benefits when cheating in singular high volume ones.

## 5 Conclusion and Future Work

In this publication, the STORE reputation system has been evaluated adopting a MAS methodology. Due to STORE's trust model based on the TI taxonomy and the rich TI model itself, a dedicated MAS framework has been designed, in which the agents form VOs, selected according to their reputation vector computed by STORE. Member agents transact with a manager, receiving a pay-off based on their reputation independent productivity measure. Four classes of agents with varying adequacy for the two selected aerospace and telco scenarios have been introduced. Unfortunately, for space reasons, the detailed mapping of the agent classes to technical TI attributes, e.g. the PDF parameters, could not be provided in this paper. The simulation results show that STORE's reputation measure can not only separate polarized agents, only performing exceptionally well or badly. It also provides decision support in difficult cases, where agents are specialized in particular VO application domains and perform on similar levels, but have to be selected for application specific roles. The results show that STORE's one configurable architecture caters for a wide range of VO scenarios. With the reputation interface, that has been extended to a vector carrying besides the generalized reputation value, also the four TCs, an informed VO manager can obtain even better decision support. He expresses his trust preferences, basically his subjective opinion about a member's trustworthy behaviour, broken down as a vector with weights for the TCs. Well expressed trust preferences, capturing the VO context, lead to a better support for the partner selection and increased productivity during the VOs operation. Finally, it has been shown that STORE very well captures gradual and temporal changes in an agent's behavior.

So far, STORE was evaluated in an aerospace and telco scenario. We plan to derive settings for simulating critical events in a VO's lifetime, e.g. replacement of a misperforming member, and expect thereby further refinement of the existing settings. We also plan to simulate further VO scenarios with properties in between the presented ones, such as an eLearning VO. Further simulation results analysing first STORE's resilience against attacks on the system itself and the reputation mechanism are available. Having started with an attack model, leading to an attack classification, we identified and simulated one novel type of freeriding attack. In this scenario, STORE's ability to recognize misbehaving agents is assessed. Second, we compared STORE to Jøsang's Beta reputation system. Both results will be published separately. Since STORE explores an approach based on a richer trust model, it takes some effort to present such a comparison. Due to STORE's flexibility, a VO scenario specific configuration can be derived that e.g. reacts faster to an organisation's changes in trustworthy

behavior than the Beta system, but introduces a tradeoff to unwanted sensitivity in short term behavior. In short, STORE with a reasonable scenario specific configuration performed consistently better in the sensitivity analysis than the Beta system, but not by a large margin.

## References

1. Strader, T., Lin, F., Shaw, M.: Information structure for electronic virtual organization management. *Decision Support Systems*, 75–94 (1998)
2. Robinson, P., Karabulut, Y., Haller, J.: Dynamic virtual organization management for service oriented enterprise applications. In: *The First International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom 2005)* (2005)
3. Cruz, M.G.: *Modeling, measuring and hedging operational risk*. Wiley, Chichester [u.a.] (2003)
4. Winkler, T.J., Haller, J., Gimpel, H., Weinhardt, C.: Trust indicator modeling for a reputation service in virtual organizations. In: *The 15th European Conference on Information Systems (ECIS)* (2006)
5. Gambetta, D.: Can We Trust Trust? pp. 213–237. Blackwell, Malden (1988); reprinted in electronic edition from Department of Sociology, University of Oxford, ch. 13
6. Haller, J.: A stochastic approach for trust management. In: *International Workshop on Security and Trust in Decentralized/Distributed Data Structures (STD3S)* (2006)
7. Josang, A., Ismail, R., Boyd, C.: A survey of trust and reputation systems for online service provision. *Decision Support Systems* (August 2004)
8. Blaze, M., Feigenbaum, J., Lacy, J.: Decentralized trust management. In: *SP 1996: Proceedings of the 1996 IEEE Symposium on Security and Privacy*, Washington, DC, USA, p. 164. IEEE Computer Society, Los Alamitos (1996)
9. Egger, F.N.: *From Interactions to Transactions: Designing the Trust Experience for Business-to-Consumer Electronic Commerce* (2003)
10. Mui, L., Mohtashemi, M., Halberstadt, A.: A computational model of trust and reputation. In: *35th Annual Hawaii International Conference on System Sciences (HICSS-35)*, vol. 7, p. 188 (2002)
11. Ismail, R., Josang, A.: The beta reputation system. In: *Proceedings of the 15th Bled Conference on Electronic Commerce* (2002)
12. Teacy, W.T.L., Patel, J., Jennings, N.R., Luck, M.: *Travos: Trust and reputation in the context of inaccurate information sources* (2006)
13. Regan, K., Cohen, R., Poupart, P.: Bayesian reputation modeling in e-marketplaces sensitive to subjectivity, deception and change. In: *Twenty-First National Conference on Artificial Intelligence (AAAI)* (2006)
14. Josang, A., Haller, J.: Dirichlet reputation systems. In: *The 2nd International Conference on Availability, Reliability and Security (ARES)* (2007)
15. Mui, L., Mohtashemi, M., Halberstadt, A.: Notions of reputation in multi-agents systems: a review. In: *Proceedings of the first international joint conference on Autonomous agents and multiagent systems*, pp. 280–287. ACM Press, New York (2002)
16. Kearney, P. (ed.): *Case study scenarios- WP11 problem definition*. TrustCoM (IST Framework 6 integrated project, grant 001945) public deliverable (2004)
17. Gale, D., Shapley, L.S.: College admissions and the stability of marriage. *American Mathematical Monthly* 69, 9–14 (1962)

# Comparison of the Beta and the Hidden Markov Models of Trust in Dynamic Environments

Marie E.G. Moe, Bjarne E. Helvik, and Svein J. Knapskog

Centre for Quantifiable Quality of Service in Communication Systems,  
Norwegian University of Science and Technology,  
O.S. Bragstads plass 2E, N-7491 Trondheim, Norway  
{marieeli, bjarne, knapskog}@q2s.ntnu.no

**Abstract.** Computational trust and reputation models are used to aid the decision-making process in complex dynamic environments, where we are unable to obtain perfect information about the interaction partners. In this paper we present a comparison of our proposed hidden Markov trust model to the Beta reputation system. The hidden Markov trust model takes the time between observations into account, it also distinguishes between system states and uses methods previously applied to intrusion detection for the prediction of which state an agent is in. We show that the hidden Markov trust model performs better when it comes to the detection of changes in behavior of agents, due to its larger richness in model features. This means that our trust model may be more realistic in dynamic environments. However, the increased model complexity also leads to bigger challenges in estimating parameter values for the model. We also show that the hidden Markov trust model can be parameterized so that it responds similarly to the Beta reputation system.

## 1 Introduction

Trust is a fundamental part of social and commercial relationships, both in the real-life and the virtual world. Complex dynamic environments, like the Internet, makes it extremely hard to obtain perfect information about potential interaction partners. In e-commerce and other electronic transactions and services, where the assets of interaction partners might be at risk, trust mechanisms may facilitate the decision-making process and lower the risk. Since trust management can be assumed to decrease risk, it can also be assumed that it will increase security and can be considered as a *soft security* mechanism [15]. Soft security accepts the fact that it is possible to circumvent the implemented security mechanisms, given enough time, effort and money. Since we might have users with malicious intentions in a system, the challenge is to detect them and find a way to monitor their behavior and possibly influence their actions, in order to prevent them from causing any harm. Trust management serves this purpose by evaluating the trustworthiness of users and offering different service levels to users based on a trust policy. If services are denied to untrusted users, an incentive for users not to misbehave, is created.

Computational trust and reputation models seek to quantify trust as a value derived from previous direct experiences and/or second-hand information, such as recommendations, and suggest mathematical and logical expressions for how to combine several

opinions about trustworthiness into reputation values. Such models are clearly needed in the virtual world where non-human agents are making trust-based decisions. But also when the human end-user is making the decisions, such calculated trust values can be very useful as decision support. For this reason a number of different trust models have been proposed. The modeling complexity varies, ranging from very simple eBay-like models to more sophisticated models based on probability theory, e.g. the Bayesian trust and reputation models [12][7][2][13][6].

In this paper we will present a comparison of our previously proposed hidden Markov trust model [11] to a binomial Bayesian reputation system [7]. The comparison is done with the help of simulations of trust scenarios. The objectives of this paper is to discuss probabilistic measurement of trust, outline the models and compare their performance in a dynamic environment where the (un)trusted objects may change behavior. We show that the hidden Markov trust model performs better when it comes to the detection of changes in behavior of agents, this means that our trust model may be more realistic for dynamic environments. We also show that the hidden Markov trust model can be parameterized so that it responds similarly to the Bayesian reputation system.

The remainder of this paper is organized as follows. Section 2 discusses the challenges related to modeling dynamic trust and how the different models can be evaluated. Section 3 gives a brief introduction to Bayesian trust models, in particular the Beta reputation system, Section 4 discusses the hidden Markov trust model, the simulation results are presented in Section 5 and Section 6 discusses the simulation results and concludes the paper.

## 2 The Dynamic Trust Modeling Problem

Since trust and reputation are active fields of ongoing research, numerous different models for quantification and evaluation of trust have been proposed. A review on some of these computational trust models can be found in [17]. However, there seems to be no single agreed upon model that can be used for benchmarking and comparison of the different trust and reputation algorithms.

Reputation models used for electronic commerce are often based on very simple mathematical formulas for combining opinions. One example is the reputation system implemented in eBay, where a feedback score is calculated as a sum of ratings that can be either positive, corresponding to a value of +1, negative with a value -1, or neutral with 0 value. A survey of trust and reputation systems that are currently used in online services can be found in [8].

In [4], several desirable qualities of reputation systems are listed. According to the authors a reputation system should be efficient, robust against attacks, easily understandable and verifiable. It should also be *weighted toward current behavior*, meaning that it responds quickly to changes in behavior so that an entity which has performed well consistently over a long time but then suddenly changes its behavior will be detected and maybe no longer trusted. This feature is missing in many trust and reputation models as trust is modeled as a static property, not taking the time component into consideration. For some applications of reputation and trust the time dependency and response to dynamic behavior are very important, as the behavior of agents could



be assumed to be highly dynamic. One example of such an application is when trust metrics are used in ad hoc routing protocols to counter malicious nodes, see for instance [9][3][2][1]. The common approach is that every node in the network monitors its neighbors and measures the frequency of packet dropping, misrouting and other potentially malicious behavior, and keeps a trustworthiness rating or reputation value recorded for all other nodes based on these observations. The underlying routing protocol is then modified with a trust component which selects routing paths and makes routing decisions based on the reputation values.

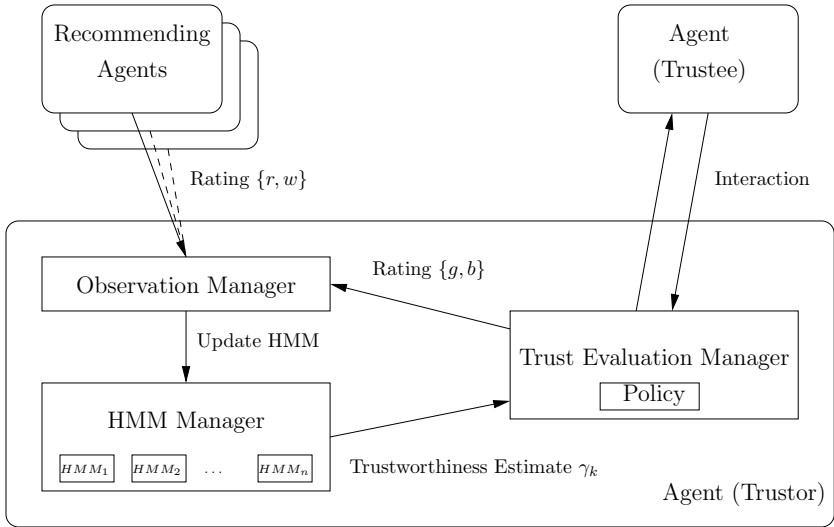


Fig. 1. The architecture of a reputation system using hidden Markov trust modeling

The computational trust algorithm used to calculate the reputation value varies. In [3], a simple eBay-like scheme is used, where reputation is a sum of recommendations of +1 whenever a packet reaches its destination, and -1 if a packet is dropped, a node is considered untrusted if its reputation value falls below a certain threshold. In [2] a more sophisticated scheme, based on a Bayesian reputation system, is used. A trust-based ad hoc routing protocol based on hidden Markov modeling of trust was proposed by the authors in [10]. The architecture of the trust component used in this approach, presented as a more general decentralized reputation system, is illustrated in Figure 1. Every agent in the system keeps and updates hidden Markov models (HMMs), that are modeling the trust state of all the other agents. Before an agent (trustor) initiates an interaction with another agent (trustee), it looks up the trustworthiness value derived from the HMM belonging to the trustee. The trustor then decides according to a policy whether or not to interact with the trustee. After an interaction, the HMM belonging to the trustee is updated with a rating, good  $g$ , or bad  $b$ , based on the outcome of the interaction. The HMMs are also updated with observations in the form of ratings from other agents in the system, so called second-hand opinions, which are either in the form

of recommendations  $r$ , or warnings  $w$ . In this study we do not include trust transitivity between different contexts, for the simplicity of the comparison, so we assume that the HMMs are only updated with observations related to the same context. We also do not consider chains of recommendations.

With this paper we would like to compare our hidden Markov modeling of trust to the Bayesian approach, in particular with regard to performance of the modeling of the dynamic aspect of trust, since this is a very important feature for applications in dynamic networking environments. A quantitative approach to comparing trust models can be found in [13]. In this paper it is proposed to use the information theoretical measure *relative entropy*. However, this approach is only applicable to probabilistic trust models that share the same fundamental assumptions about the underlying probability distributions. In cases where a direct mathematical comparison of models is difficult, comparison by the help of simulations seems to be the most viable approach. Different trust scenarios can be simulated in order to see which trust and reputation system performs best with regard to reliability of the calculated values under various hostile agent strategies.

### 3 Bayesian Trust Modeling

Bayesian trust models, for calculating reputation scores from ratings, are based on the assumption that the behavior of an agent can be described according to a probability distribution. The trust value is a function of the expected value of the probability distribution, which gets updated with every new rating received according to Bayes' Theorem. Binomial Bayesian reputation systems, where ratings can be expressed by two values, *good* or *bad*, are modeled with the Beta probability density function [12][7][2]. Multinomial Bayesian reputation systems, that allow for ratings with graded levels, are modeled with the Dirichlet probability density function [13][6]. In this paper we will focus on the binomial case, and evaluate the performance of our proposed trust model compared to the Beta reputation system proposed by Jøsang et al. [7].

#### 3.1 The Beta Reputation System

The Beta reputation system models the reputation formation for a trustor as a sequence of observations, where each observation is the outcome of the rating done by a trustee, based on the outcome of an interaction. A reputation centre collects ratings from all the agents, and updates each agent's reputation score.

The underlying mathematical model of the Beta reputation system considers the ratings as a sequence of trials with binomial outcomes, for each trial there is a probability  $p$  of getting a *good* rating (recommendation) and a probability  $(1 - p)$  of getting a *bad* rating (warning). The parameter  $p$  belonging to a trustor is initially unknown, so due to lack of information it is assumed that it is drawn from a uniform distribution on  $[0, 1]$ . As ratings concerning this trustor start to arrive, there is more information available and we can update the distribution of  $p$ . In accordance with Bayesian inference we have a *prior* hypothesis  $X$  about the outcome of a trial, which is updated *a posteriori* to the actual outcome  $Y$  in accordance with Bayes' Theorem

$$P(X | Y) = \frac{P(X)P(Y | X)}{P(Y)}. \tag{1}$$

The Beta distribution

$$Beta(\alpha, \beta) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} p^{\alpha-1} (1 - p)^{\beta-1} \tag{2}$$

is a *conjugate prior* for binomial trials (Bernoulli process). This means that if we assume that the prior  $X$  hypothesis is described by  $Beta(\alpha, \beta)$ , and  $Y$  is a sequence of ratings, out of which  $r$  is the number of good ratings (recommendations) and  $w$  is the number of bad ratings (warnings), then the posterior  $P(X | Y)$  is also described by a Beta distribution  $Beta(\alpha + r, \beta + w)$ . The initial prior is given by  $Beta(1, 1)$ , which corresponds to the uniform distribution on  $[0, 1]$ . The reputation value is given as a function of the expectation value of the Beta distribution  $E(p) = \alpha / (\alpha + \beta)$ , for the posterior hypothesis the expectation is found by setting  $\alpha = r + 1$  and  $\beta = w + 1$ . This results in a very simple calculation of the probability expectation value. Let  $(r_k, w_k)$  denote the ratings received at iteration step  $k$ , we then get the following recursion for deriving the Beta parameters:

$$\alpha_k = \alpha_{k-1} + r_k, \quad \beta_k = \beta_{k-1} + w_k, \quad \alpha_0 = \beta_0 = 1. \tag{3}$$

For finding the probability expectation value at iteration step  $k$  we get:

$$E(p_k) = \frac{\alpha_k}{\alpha_k + \beta_k}. \tag{4}$$

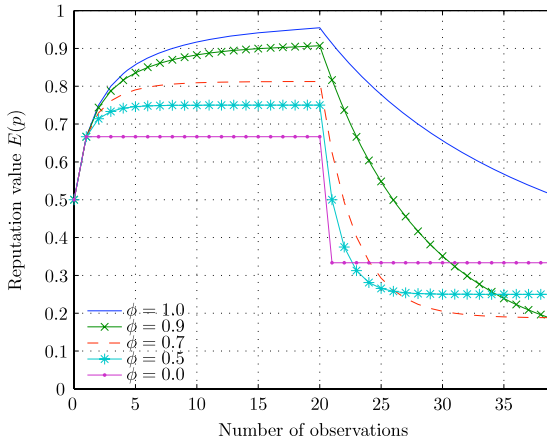
The probability expectation value given in Equation 4 gives a reputation rating in the range  $[0, 1]$ , where the value 0.5 represents a neutral reputation value. To make the reputation model more realistic, several modifications to the calculation of the reputation value are introduced. These variations include *discounting* of ratings based on the reputation of the agent providing the rating, *forgetting* old ratings by giving old ratings less weight than more recent ratings, and *weighting* of ratings according to the value of the rated transaction.

### 3.2 Evaluation of the Beta Model

The Beta reputation system without forgetting factor is efficient, easily understandable and verifiable, but it is not weighted toward current behavior. This is due to the underlying Bayesian framework, which assumes that the behavior of agents can be approximated by a *fixed* probability distribution. Since agents may change behavior over time, this static modeling is not realistic. The forgetting factor  $0 \leq \phi \leq 1$  was introduced in 7 to overcome this problem. It is used to scale the parameters  $(\alpha, \beta)$  in every update of the Beta distribution, so that the we get

$$\alpha_k^* = \alpha_{k-1}^* \phi + r_k, \quad \beta_k^* = \beta_{k-1}^* \phi + w_k, \quad \alpha_0^* = \beta_0^* = 1. \tag{5}$$

A forgetting factor  $\phi = 1$  means that all ratings are weighted equally, and nothing is forgotten, with  $\phi = 0$  only the last rating is remembered. In Figure 2 we can see how



**Fig. 2.** The Beta model with different forgetting factors, the observations are 20 good ratings followed by 20 bad ratings

the Beta model responds to a sequence with 20 good ratings followed by 20 bad ratings, with different forgetting factors.

As noted by the authors of [13], the forgetting factor is a form of exponential decay on the parameters of the Beta model giving an effective bias towards newer information, but it is unclear if this fading mechanism is really modeling dynamic behavior of the agents. If agents were likely to change their behavior in such a way that the probability  $p$  of getting good ratings slowly increases or decreases, this fading of parameters seems like a good modeling approach. However, if we consider a disruptive agent that follows a strategy where it behaves good for a certain amount of time, building up a good reputation value, and then suddenly starts to misbehave taking advantage of its reputation, this slowly adapting model might not be good enough.

Another problem with the Beta model is the lack of time component. The reputation formation is only depending on the number of ratings, without taking the time between ratings into account. If we assume that ratings are not received at regular intervals, the claim that the forgetting factor takes care of adjusting the model towards *new* information may not be valid anymore. A simple way of rectifying this is by introducing a time stamp on the ratings, like suggested in [19].

## 4 The Hidden Markov Trust Model

The hidden Markov trust model takes the time between observations into account, it also distinguishes between system *states* and uses methods previously applied to intrusion prevention [5] for the prediction of which state an agent is in. The hidden Markov trust model was originally proposed by the authors as a component in a trust-based ad hoc routing protocol [10]. It was further developed with a parameter learning component for multiagent environments [11].

## 4.1 Hidden Markov Modeling

A hidden Markov model (HMM) consists of a finite set of  $N$  hidden states  $S = \{s_1, \dots, s_N\}$  with an associated probability distribution. The state of the monitored agent is described by a discrete time Markov chain  $\mathbf{x}_k = x_1, x_2, \dots$  where  $x_k \in S$  is the possibly hidden state of the agent at sampling instant  $k$ .  $\mathbf{P}_k = \{p_{ij}^k\}$  is the set of state transition probabilities,  $p_{ij}^k = P(x_{k+1} = s_j \mid x_k = s_i)$ ,  $1 \leq i, j \leq N$ , where  $x_k$  is the current state of the system.  $\pi = \{\pi_i\}$  is the initial state distribution, where  $\pi_i = P(x_1 = s_i)$ ,  $1 \leq i \leq N$ . The output from the agent ratings is classified by the set of observation symbols  $V = \{v_1, \dots, v_M\}$ . Let  $\mathbf{y}_k = y_1, y_2, \dots$  denote the sequence of observations, where  $y_k \in V$  is the observation made at sampling instant  $k$ . The HMM consists of two stochastic processes; the hidden process  $\mathbf{x}_k$ , and the observable process  $\mathbf{y}_k$  that depends on  $\mathbf{x}_k$ . The relation between  $\mathbf{x}_k$  and  $\mathbf{y}_k$  is described by the probability distribution matrix  $\mathbf{B} = \{b_j(m)\}$ , where  $b_j(m) = P(y_k = v_m \mid x_k = s_j)$ , for  $1 \leq j \leq N$ ,  $1 \leq m \leq M$ . See for instance [14] for a more extensive introduction to HMMs.

In the hidden Markov trust model considered in this paper, we choose to use two hidden states  $\{trusted, untrusted\}$ , and four observation symbols  $\{g, b, r, w\}$ , corresponding to the observations *good*, *bad*, *recommendation* and *warning*. The reason for choosing two states is to make it easier to compare the model with the binomial Bayesian model. A comparison of our model with more states and more observation symbols to a multinomial Bayesian model would be an interesting topic for our future work. An agent is in an untrusted state if it has been behaving in a malicious way in previous interactions, it is in a trusted state if it has shown good behavior. We model trust as a dynamic variable, changing with time. This allows us to capture the behavioral characteristics of agents that are behaving good for a certain time, but then suddenly start misbehaving. Since an agent's behavior can be changing with time it is not necessarily the case that an agent is in the same state as it were at the last encounter. An agent can only do its best guessing about the trustworthiness state of an other agent based on its own previous direct experiences, which were either *good* or *bad*, and *recommendations* or *warnings* from other agents in the system. This means that the system state is hidden, and hence we use the HMM approach.

We consider a decentralized reputation system where each agent updates its own trust value for the other agents based on its own direct experiences, and from feedback in the form of ratings communicated from other agents in the multiagent system. We model the agent interaction as a stochastic process. This means that we assume that there is a random time interval between each agent interaction and that the behavior of an agent is only dependent on its current state. When using a Markov model to model the state of an agent, we make the following assumptions; all information about the agent is contained in the state, observations are independent given the current state, and state occupation time is negatively exponentially distributed.

## 4.2 State Probability Distribution

From the HMM we can derive a prediction of the probability distribution over the states, and we use the probability of being in the trusted state as reputation value. Our modeling approach is different from the Beta model as we do not assume that there is an

underlying fixed probability  $p$  of getting a good rating. Instead we assume that an agent is in one of the hidden states, and that the ratings are characterized by different values of  $p$  dependent on the current state of an agent. The rating process is similar to the monitoring process in an intrusion detection system, and the challenge is to predict the current state of an agent and detect a possible state change.

We have not made any assumptions about time between observations, and there is no direct relation between observations and state-changes. As a consequence the system could have made zero, one or more transitions during the time between to successive observations. The time when observation number  $k$  is produced is denoted  $t_k$ . Time between observation  $k - 1$  and observation  $k$  is denoted  $\delta_k = t_k - t_{k-1}$ .

The transition rate matrix  $\Lambda = (\lambda_{ij})$  is describing the dynamics of the system. To simplify the notation we will use  $i$  and  $j$  instead of  $s_i$  and  $s_j$ . The relation between system states and the transition rates is given by

$$\lambda_{ij} = \begin{cases} \lim_{dt \rightarrow 0} \frac{P(\mathbf{x}(t+dt) = j | \mathbf{x}(t) = i)}{dt} & \text{if } i \neq j \\ \sum_{j \neq i, j=1}^N -\lambda_{ij} & \text{if } i = j \end{cases}. \quad (6)$$

Since observations are received at irregular intervals, the running transition probabilities  $p_{ij}^k = P(x(t + \delta_k) = j | x(t) = i)$  depend on the time since last observation  $\delta_k$ , and have to be calculated each time an observation is received. The running transition probability matrix  $\mathbf{P}_k = (p_{ij}^k)$  can be derived from Kolmogorov's equations [16] as follows

$$\mathbf{P}_k = e^{\Lambda \delta_k}. \quad (7)$$

There are several analytical and numerical methods for solving these ordinary differential equations, in our case the state space is very small, so calculations are inexpensive. Let  $\gamma_k = (\gamma_k(i))$  denote the prediction of the state probability distribution at time  $t_k$  given all observations received until time  $t_k$ ,  $\gamma_k(i) = P(x_k = i | \mathbf{y}_k)$  where  $\mathbf{y}_k = y_1, \dots, y_k$ . The algorithm for calculating  $\gamma_k$  is given in [5], it is an on-line algorithm derived from the *forward-backward* procedure described in [14], and is very efficient. It does not require the agents to keep any history of past observations in memory.

### 4.3 Parameter Estimation

The parameters that need to be set in the HMM are the initial state distribution  $\pi$ , the observation symbol probabilities  $\mathbf{B}$  and the state transition rates  $\Lambda$ . In [11] we describe a method for learning the model parameters by the combination of the machine learning technique *reinforcement learning* [18] and the forward-backward procedure, which finds the maximum likelihood parameter estimate from a training sequence of observations. As this parameter learning technique is not the main focus of this paper, we will assume that these parameters are available to the model and perform the simulations with a few different representative values for the parameters.

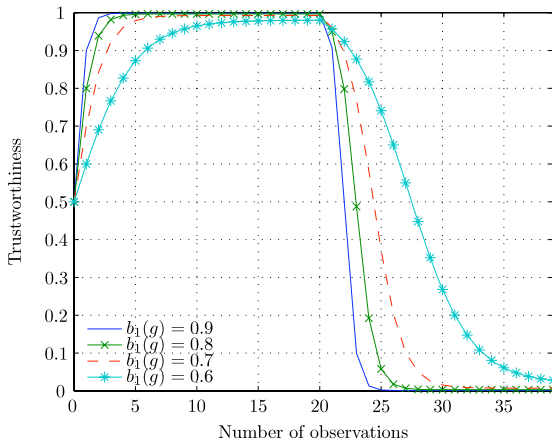
We will set the initial state distribution  $\pi$  to be uniform over the states, so we have that  $\pi_1 = 0.5$  and  $\pi_2 = 0.5$ , in order to get the same starting condition as the Beta model. However, to overcome the problem of agents changing their identities and re-entering the system frequently, it might be better to change the starting condition so that a newcomer to the system is most likely not in a trusted state.

The state transition rates can be calculated from estimated expected state sojourn times  $H = (h_1, h_2)$ , the relation between transition probabilities and transition rates is given by

$$\lambda_{ij} = \frac{P_{ij}}{h_i} \quad \text{for } i \neq j. \tag{8}$$

The transition rate models the tendency of the agent to change its trustworthiness over time, large state transition rates will lead to faster response to indications of state changes in the model.

The observation symbol probabilities models the uncertainty of the observations. If we for instance have the parameter  $b_1(g) = 0.9$ , this means that we have a probability of 0.1 of getting a good observation even though the agent really is in an untrusted state. In other words we have a certainty of 90% of getting correct observations. Figure 3 shows how the hidden Markov trust model responds to an input of 20 good followed by 20 bad observations for different observation symbol probabilities. The time between observations is fixed, and we have used the estimated state sojourn times  $h_1 = 100$ , and  $h_2 = 100$ . We have used symmetric observation probabilities in this example, i.e. if  $b_1(g) = 0.9$  we also have that  $b_2(b) = 0.9$ .



**Fig. 3.** The hidden Markov trust model with different observation probabilities, the input is 20 good direct observations followed by 20 bad direct observations

As we can see from Figure 3, the observation symbol probabilities influence the response to state transitions in the model. This is natural, since if the observations are unreliable, we would like to have more observations indicating a state change before we believe that an actual state change has occurred. It would make sense to assign a higher observation symbol probability to the first-hand observations  $\{g, b\}$  than to the second-hand observations  $\{r, w\}$ . For the second-hand observations we could choose to model each recommender separately, this means that we assign different observation symbol probabilities  $\{r, w\}$  to every recommender. If we have a history of previous

recommendations and warnings coming from a specific recommender, we can learn the parameters from this sequence of observations.

## 5 Simulation Results

In this section we will present some simulation results from our comparison study of the hidden Markov trust model and the Beta reputation model. We describe a selection of trust scenarios and compare the performance of the models in these situations.

### 5.1 Simulation Assumptions and Parameters

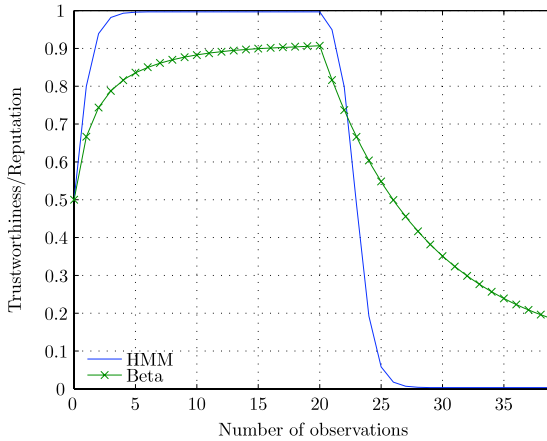
When we do the comparison of the Beta model and the hidden Markov trust model in the following, we will consider a decentralized version of the Beta reputation system, where we let each agent calculate its own reputation value for the other agents instead of calculating the reputation values in a reputation centre. We assume that there is a trusted reliable communication protocol in place that allows agents to obtain feedback from other agents in the form of ratings.

For the model parameters, we have used the Beta model with a forgetting factor  $\phi = 0.9$ , and the hidden Markov trust model with state sojourn times  $h_1 = 100$ ,  $h_2 = 100$  and observation symbol probabilities  $b_1(g) = 0.8$ ,  $b_2(b) = 0.8$ . For the Beta model, we can see from Figure 2 that a high value of  $\phi$  gives the best response to state changes as it gives the largest variation in the reputation value. Small values of  $\phi$  seems to give quicker response, but leads to a convergence of the reputation score to a less extreme value. This means that the reputation value becomes more average and does not clearly distinguish between states. Since we want to model these state changes with our hidden Markov trust model, we have used the Beta model with a high value of  $\phi$ . For the hidden Markov trust model, we have used relatively high observation probabilities following the same reasoning. In the last simulation we have used other parameters for the hidden Markov trust model, because we want to illustrate the flexibility of the model by showing how we can adjust the parameters so that it responds similarly to the Beta model.

### 5.2 Response to State Changes

We have already seen from Figures 2 and 3 how the two models respond to a state change, we have 20 good observations followed by 20 bad observations. In Figure 4 we see the difference between the models more clearly. Such an input set of observations could come from a trust scenario where an agent builds its reputation value by behaving good for a certain amount of time, and then decides to take advantage of its good reputation by suddenly changing its behavior. From Figure 4 we see that the slope of our model is much steeper than the slope of the Beta model. The Beta model has a lower reputation value for the first observation, but this is due to the slower convergence of the Beta model to the good state. If we for instance had a threshold for detecting state changes at the reputation value 0.5, the hidden Markov trust model would detect this already at the third bad observation while the Beta model would detect it after six bad observations.





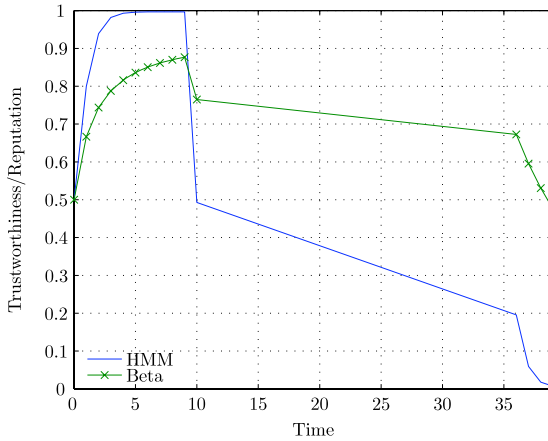
**Fig. 4.** The hidden Markov trust model compared to the Beta model, the input is 20 good observations followed by 20 bad observations

### 5.3 Time Component

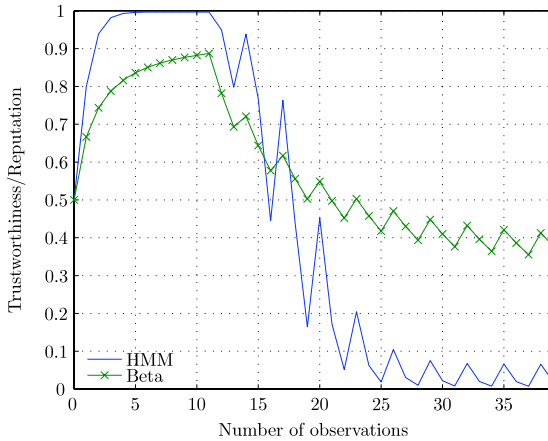
The Beta model does not take the time component into consideration, it only models the reputation value in terms of number of ratings. In the hidden Markov trust model we include the time between observations in our model. To illustrate the advantage of including the time aspect, we consider the following scenario. We assume that an agent has been compromised, i.e. 'taken over' by a malicious agent. The agent then proceeds with a strategy of 'laying low', meaning that it waits for a long time without acting malicious, so that when it starts to show malicious behavior it can take full advantage of the good reputation that the previous owner of the agent had built up. From Figure 5 we can see an example of such a scenario, where we have 9 good observations, then one bad observation at time  $t = 10$ , then no observations until time  $t = 35$ , followed by 5 bad observations. We can observe from the plot that the hidden Markov trust model gives a steeper slope and continues the negative trend over time, while the Beta model is just stretched at the  $x$ -axis.

### 5.4 Disruptive Behavior

We want to see how the models react to a disruptive agent that changes its strategy in order to adapt to the rules of threshold-based intrusion detection. In particular, we consider an agent that follows a pattern of misbehavior adapted to a detection rule of 'three strikes and you're out'. In Figure 6 we have an example of this scenario, where an agent is showing good behavior for 10 observations to build up its reputation, and then proceeds with the disruptive behavior giving a pattern of 2 bad observations, one good observation, 2 bad observations, and so on. We can see from the plot that the Beta model picks up this behavior with a decreasing reputation value, but the hidden Markov trust model detects the state change faster and converges to much lower trustworthiness values.



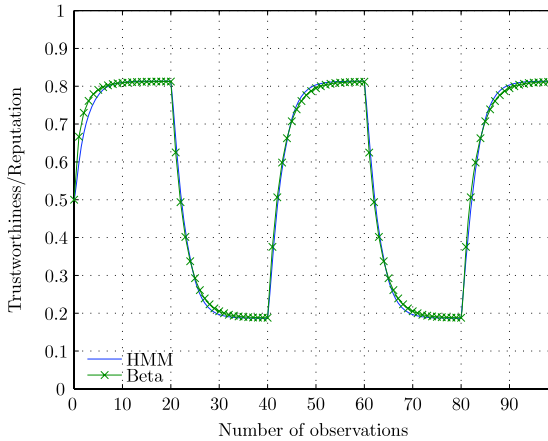
**Fig. 5.** The hidden Markov trust model compared to the Beta model, the input is 9 good observations, 1 bad observation at time  $t = 10$ , then no observations until time  $t = 35$  followed by 5 bad observations



**Fig. 6.** The hidden Markov trust model compared to the Beta model, the input is 10 good observations, followed by a disruptive behavior giving a pattern of 2 bad observations, one good observation, 2 bad observations, and so on

### 5.5 Model Flexibility

We have shown some examples where the hidden Markov trust model performs better than the Beta model in detecting state changes. This is not very surprising as the Beta model is not based on the assumption that agents can be in different *states* when it comes to trustworthiness. The performance of both models is of course dependent on the model parameters. The Beta model in the variant that we used in our simulations has fewer parameters than the hidden Markov trust model, albeit we have included the



**Fig. 7.** The hidden Markov trust model compared to the Beta model, the parameters of the models have been adjusted to make them respond similarly, input is 20 good observations followed by 20 bad observations, 20 good observations and so on

forgetting factor as a parameter in order to study the variant of the Beta model which is most sensitive to dynamic behavior. We used the parameters that seemed to give the most beneficial results for both models.

Now we want to illustrate the flexibility of the hidden Markov trust model, by adjusting its parameters so that it responds similarly to the Beta model. The results of this adjustment can be seen in Figure 7. We have used the Beta model with a forgetting factor of  $\phi = 0.7$ , and adjusted the parameters of our model to make it respond close to the Beta model. For the hidden Markov trust model we have used the observation symbol probabilities  $b_1(g) = 0.6$ ,  $b_1(b) = 0.4$ ,  $b_2(b) = 0.6$  and  $b_2(g) = 0.4$ . We also adjusted the state sojourn times to  $h_1 = 8$  and  $h_2 = 8$ . The hidden Markov trust model with these parameters describes a situation where observations are very uncertain and state transition rates are high. With such parameters we could say that the state modeling aspect of our model has been suppressed.

## 6 Discussion and Conclusion

We have seen from the simulated examples that the Beta model and the hidden Markov trust model performs differently. We will now explain the fundamental differences between the two models, and discuss the findings from the simulations in this light. The hidden Markov model assumes an underlying state, observations are uncertain and we have an uncertainty of which state an agent is in. The Beta model does not assume that an agent is either good or bad, but rather seeks to pinpoint the trustworthiness of an agent on a continuous scale from 0 to 1. The interpretations of the observations in this model are deterministic. The difference between the models can be seen as an analogy to the difference between fuzzy sets and probabilities. In fuzzy logic an agent can be

partially trusted, in the sense that he is 70% honest and 30% dishonest. This is different from a situation where we are 70% certain that an agent is 100% honest. This fundamental difference between the two models explains why the hidden Markov model performs better when it comes to the detection of changes in behavior of the agents over time. While the hidden Markov model recognizes a state transition, the Beta model is instead modeling an agent that gradually becomes partially more dishonest. This difference is clearly demonstrated in the simulation illustrated in Figure 6, where we consider an agent with a disruptive strategy. Additionally, we have the effect of the different time constants in the models. While the Beta model is relying on the 'lifetime' of old observations, the time constant in the hidden Markov trust model is associated with the underlying state transition process.

The hidden Markov trust model has more parameters than the Beta model, thus it can be more fine-tuned and adaptable to dynamic environments. However, this also leads to challenges related to the parameter estimation. In [11] it is discussed how its parameters can be learned using a combination of the machine learning technique *reinforcement learning* [18] and the forward-backward procedure [14], which finds the maximum likelihood parameter estimate. Both the Beta model and the hidden Markov trust model can be further refined by introducing more dimensions or states. The multinomial Bayesian models, which allow for graded ratings, introduce more dimensions to the Bayesian modeling. It would have been interesting comparing a multinomial Bayesian trust model to a hidden Markov trust model with more states and more observation symbols. However, such a comparison would be challenging due to the big number of parameters that would need to be managed in the simulations. Including trust transitivity between different contexts is also an important issue that should be addressed in future work.

We have presented a comparison of the hidden Markov trust model and the Beta reputation system. Due to its larger richness in model features, the hidden Markov trust model shows a better ability to deal with dynamic environments, where we are unable to obtain perfect information and agents can be assumed to change their behavior over time. However, the increased model complexity also leads to larger challenges in finding representative parameters for the model. A disadvantage of both models might be that they are not easily understandable to human users, since they build on much more advanced mathematics than the simple eBay-like systems. These models are therefore maybe better suited for applications in multiagent systems, routing protocols and other distributed networking environments with non-human interpreters of the trustworthiness calculations.

## References

1. Balakrishnan, V., Varadharajan, V., Tupakula, U., Lucs, P.: TEAM: Trust Enhanced Security Architecture for Mobile Ad-hoc Networks. In: ICON 2007: 15th IEEE International Conference on Networks, pp. 182–187. IEEE, Los Alamitos (2007)
2. Buchegger, S., Le Boudec, J.: A Robust Reputation System for P2P and Mobile Ad-hoc Networks. In: Proceedings of the Second Workshop on the Economics of Peer-to-Peer Systems (2004)

3. Dewan, P., Dasgupta, P., Bhattacharya, A.: On using reputations in ad hoc networks to counter malicious nodes. In: ICPADS 2004: Proceedings of the Tenth International Conference on Parallel and Distributed Systems. IEEE, Los Alamitos (2004)
4. Dingedine, R., Freedman, M., Molnar, D.: Accountability. In: Oram, A. (ed.) *Peer-to-Peer: Harnessing the Power of Disruptive Technologies*. O'Reilly, Sebastopol (2001)
5. Haslum, K., Moe, M.E.G., Knapskog, S.: Real-time Intrusion Prevention and Security Analysis of Networks using HMMs. In: Proceedings of the Fourth IEEE LCN Workshop on Network Security (WNS 2008). IEEE, Los Alamitos (2008)
6. Jøsang, A., Haller, J.: Dirichlet Reputation Systems. In: Proceedings of the Second IEEE International Conference on Availability, Reliability and Security (ARES 2007). IEEE, Los Alamitos (2007)
7. Jøsang, A., Ismail, R.: The Beta Reputation System. In: Proceedings of the 15th Bled Electronic Commerce Conference (2002)
8. Jøsang, A., Ismail, R., Boyd, C.: A survey of trust and reputation systems for online service provision. *Decision Support Systems* 43(2), 618–644 (2007)
9. Marti, S., Giuli, T.J., Lai, K., Baker, M.: Mitigating routing misbehavior in mobile ad hoc networks. In: *MobiCom 2000: Proceedings of the 6th annual international conference on Mobile computing and networking*, pp. 255–265. ACM, New York (2000)
10. Moe, M.E.G., Helvik, B.E., Knapskog, S.J.: TSR: Trust-based Secure MANET Routing using HMMs. In: Proceedings of the 4th ACM International Symposium on QoS and Security for Wireless and Mobile Networks (Q2SWinet 2008). ACM, New York (2008)
11. Moe, M.E.G., Tavakolifard, M., Knapskog, S.J.: Learning Trust in Dynamic Multiagent Environments using HMMs. In: Proceedings of the 13th Nordic Workshop on Secure IT Systems (NordSec 2008) (2008)
12. Mui, L., Mohtashemi, M., Halberstadt, A.: A Computational Model of Trust and Reputation. In: Proceedings of the 35th Annual Hawaii International Conference on System Sciences, pp. 2431–2439 (2002)
13. Nielsen, M., Krukow, K., Sassone, V.: A Bayesian Model for Event-based Trust. *Electronic Notes on Theoretical Computer Science (ENTCS)* 172, 499–521 (2007)
14. Rabiner, L.R.: A tutorial on hidden markov models and selected applications in speech recognition. In: *Readings in speech recognition*, pp. 267–296 (1990)
15. Rasmusson, L., Jansson, S.: Simulated Social Control for Secure Internet Commerce. In: Proceedings of the 1996 Workshop on New Security Paradigms (NSPW 1996). ACM, New York (1996)
16. Ross, S.M.: Continuous-Time Markov Chains. In: *Introduction to Probability Models*, 8th edn., pp. 349–390. Academic Press, London (2003)
17. Sabater, J., Sierra, C.: Review on computational trust and reputation models. *Artificial Intelligence Review* 24(1), 33–60 (2005)
18. Sutton, R., Barto, A.G.: *Reinforcement Learning: An Introduction*. MIT Press, Cambridge (1998)
19. Whitby, A., Jøsang, A., Indulska, J.: Filtering out unfair ratings in bayesian reputation systems. In: Proceedings of the 7th International Workshop on Trust in Agent Societies (2004)

# WRS: The Wikipedia Recommender System

Thomas Lefèvre, Christian Damsgaard Jensen, and Thomas Rune Korsgaard

Informatics and Mathematical Modelling  
Technical University of Denmark  
Christian.Jensen@imm.dtu.dk

**Abstract.** In 2005, the Wikipedia became the most popular reference website on the Internet and it has continued to grow in size and popularity ever since. With the increasing reliance on the Wikipedia comes issues of the credibility and provenance of content. In order to address these issues, we have developed a Recommender System for the Wikipedia, which allows the users of the Wikipedia to rate articles in order to guide other users about the quality of articles. This rating system provides both an incentive for authors to improve articles and a quantifiable measure of the perceived quality of articles.

## 1 Introduction

The Wikipedia is an online encyclopedia that is collaboratively edited by users on the Internet. The Wikipedia's philosophy is that anyone who wishes to share their knowledge about a subject can edit the article on that subject. The process through which Wikipedia content is added and modified is largely unregulated, which has raised concerns about the credibility of the Wikipedia [12] and there are plenty of examples of erroneous information that has propagated through the Wikipedia [34]. It is, however, apparent that the Wikipedia has gained the trust of the Internet population, despite the fact that there is nothing inherently trustworthy about the Wikipedia. It is therefore important to provide Wikipedia users with a simple and intuitive way to assess the trustworthiness of the content they are reading. We have therefore developed a recommender system, which provides users with an assessment of the quality of Wikipedia articles based on the feedback from other users who read the same article.

In order to preserve backward compatibility with the existing architecture, the Wikipedia Recommender System (WRS) is implemented as a proxy between the Wikipedia site and the user's browser. We have based the WRS on an extensible web-proxy technology called Scone [8]. The Scone proxy intercepts HTML documents, among other things, tokenizes them and allows user developed plugins to manipulate these tokenised documents before they are handed over to the user's browser. The WRS is implemented as a plugin to Scone, which makes it possible to intercept Wikipedia requests and collect recommendations from other users about the article and present an aggregated value that represents these recommendations to the user through the WRS user interface that is injected into the browser. The user can then use this interface to read estimated ratings as well as rate articles himself. The general architecture of the WRS is shown in Figure 1.

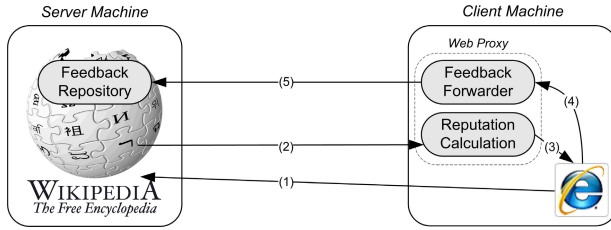


Fig. 1. Architecture of the Wikipedia Recommender System

When a user requests a Wikipedia article (1), the Reputation Calculation module extracts the rating data from Wikipedia (2). It then uses this data to calculate an aggregated rating of the article which is passed to the user interface embedded into the web page (3). By using the interface (4), the user can rate the article and the Feedback Forwarder will compile the necessary data, and upload it to the appropriate location on Wikipedia (5). The Feedback Repository, which stores the recommendations from WRS users and the WRS client proxy are described in greater details in the following.

## 2 Feedback Repository

A Recommender System needs to store all the recommendations in a place that is accessible at all times, by all users. The storage should also require the least amount of work to setup, maintain and operate compared to its target environment. Since the Wikipedia allows user editing, it is possible to use this mechanism to store meta-data, such as recommendations, inside the Wikipedia articles, e.g., as HTML comments that will not be rendered by the browser. This makes the WRS compatible with all Wikis based on the MediaWiki software, not just the Wikipedia.

Storing the recommendations inside the existing Wikipedia means that implementation of the WRS required no modifications to the MediaWiki software and it requires no additional maintenance or operation procedures on the Wikipedia servers. Moreover, recommendations are available to all users who have access to the Wikipedia, because they are stored on the same servers. The first part of the recommendation contains the rating of the article which is readable by humans, but the second part is a BASE64 encoded digital signature by the user who submitted the rating.

The meta-data associated with the WRS can be stored 'invisibly' in articles via the wikitext markup language, which makes it easy to store the meta-data anywhere on the Wikipedia. However, storing the meta-data inside Wikipedia articles causes problems, because it results in longer load times for all users, not only the users of the WRS. This is not considered good behavior by ordinary Wikipedia users, so another solution is needed. Fortunately, registered users are given their own user-page, which is part of the Wikipedia and can be edited by everybody. We have therefore created the user `recommendations` which stores all the recommendations for all users of the WRS. We have created sub-pages in this user's user-page, which mirrors the rated articles from

the main Wikipedia, but these sub-pages only contain the recommendations. As an example this naming convention, consider the article [http://en.wikipedia.org/wiki/Bass\\_Strait](http://en.wikipedia.org/wiki/Bass_Strait), which becomes [http://en.wikipedia.org/wiki/User:Recommendations/Bass\\_Strait](http://en.wikipedia.org/wiki/User:Recommendations/Bass_Strait)

### 3 WRS Client Proxy

The client proxy consists of two components, the Feedback Forwarder and the Reputation Calculator. The Feedback Forwarder prepares the user recommendation and signs it, then prepares the meta-data and uploads it in the correct location in the Wikipedia. The Reputation Calculator extracts and calculates on this meta-data, which will be explained in further detail in the following.

When the user requests a Wikipedia article, the Reputation Calculator accesses the Feedback Repository and extracts the recommendations. It then verifies the recommendations, making sure that they are valid and properly signed. The Reputation Calculator then aggregates the recommendations for the article, based on the ratings in the recommendations and the requesting user’s trust in the recommenders. The aggregated rating is then presented to the user through the WRS user interface shown in Figure 2. The WRS user interface shows the aggregated quality indication at the top (the value “3.0” in Figure 2). Below this line, the user is asked whether he agrees with this value (“Was This Information Useful to You?”) and the user is asked to provide his own rating at the bottom of the window. The usefulness indicator and the user’s own rating are used to calculate the user’s trust in the recommenders.

The WRS Client Proxy maintains a database of all users who have rated articles accessed by the user. When the user then rates an article on the Wikipedia, the Reputation Calculator compares this rating with the recommendations received from other users. Recommendations that contain ratings close to the user’s own rating are considered positive and result in a higher trust in these recommenders. Similarly, recommendations with ratings that are very different from the user’s own rating will be considered negative and the user’s trust in those recommenders will decrease. The user’s trust in



Fig. 2. The WRS user interface



the recommender is used together with the rating in the recommendation to calculate the aggregated article rating. If the user's trust in a recommender is high, the rating will contribute more to the aggregate rating. This means that the value of the WRS to an individual user depends on the user's rating of articles, which build trust in the other users as recommenders, but at the same time contribute recommendations to the WRS community.

## 4 Discussion

The WRS addresses the important problem of providing Wikipedia users with essential information about the credibility of Wikipedia articles. The developed prototype of the WRS integrates seamlessly with the existing Wikipedia infrastructure in a way the benefits users of the WRS without penalising Wikipedia users who do not use the WRS.

## References

1. McHenry, R.: The Faith-Based Encyclopedia. TCSDaily.com (November 15, 2004) (visited March 14, 2009)
2. Denning, P., Horning, J., Parnas, D., Weinstein, L.: Wikipedia Risks, in *Inside Risks* 186. Communications of the ACM 48(12) (2005)
3. Seigenthaler, J.: A false Wikipedia 'biography'. USA TODAY (November 29, 2005)
4. Orłowski, A.: Avoid Wikipedia, warns Wikipedia chief, It can seriously damage your grades. The Register (June 15, 2006)
5. Korsgaard, T.R.: Improving Trust in the Wikipedia. M.Sc. Thesis, Department of Informatics & Mathematical Modelling, Technical University of Denmark (2007)
6. Korsgaard, T.R., Jensen, C.D.: Reengineering the Wikipedia for Reputation. In: Proceedings of the 4th International Workshop on Security and Trust Management (STM 2008), Trondheim, Norway, pp. 71–84 (2008)
7. Jensen, C.D., Korsgaard, T.R.: Dynamics of Trust Evolution: Auto-configuration of dispositional trust dynamics. In: Proceedings of the International Conference on Security and Cryptography (SECRYPT 2008), Porto, Portugal (2008)
8. Scone website, <http://www.scone.de> (visited March 15, 2009)

# Distributed Systems Security Governance, a SOA Based Approach

Pierre de Leusse<sup>1,2</sup> and David Brossard<sup>2</sup>

<sup>1</sup> Newcastle University

<sup>2</sup> BT Innovate

Pierre.de-leusse@ncl.ac.uk

## 1 Introduction

The aim of this demonstration is to show how a governed composition of security related services, provided through the Security as a Service (SaaS) paradigm, can be leveraged on in order to provide a more flexible and usable approach to security in distributed and complex systems.

The demonstration will feature the presentation of a security governance gateway that allows manipulating the security configuration of resources exposed through it in a more dynamic way compared to existing techniques. An additional key aspect of the governance gateway is to improve the visibility of the different parameters to take in account when securing the access to a resource in order to make the decision process more adequate. Through this presentation, the demonstrators hope to show the practicability and interest of governed, composable and adaptable security.

## 2 Objectives of SOA Security Governance

Functional decomposition into services, reuse, loose coupling, and distribution of resources are all perceived benefits of the investment on SOA. This malleability can also bring about the risk of a more difficult oversight. The same service is used in different applications the infrastructure will have to adapt to these different contexts of use in order to provide variations in required functionality, varying quality of service, varying billing schemes, and meet varying security requirements. Achieving such variations in a cost efficient way can be achieved by composing the core business function offered by a service with other services implementing infrastructure capabilities that fulfil varying non-functional requirements.

However, as the number of services increases and their use in different contexts proliferates, it becomes necessary to automate policy enforcement and compliance monitoring. Furthermore, the composition of services into different business applications over a common infrastructure intensifies the need for end-to-end monitoring and analysis to assess the business performance impact. Managing the full life-cycle of service definition, deployment, exposure and operation requires management processes that take into account their composition with the infrastructure capabilities that take of non-functional requirements. Finally, policies may change during the life-time of a service. Policy updates may be the result of various reasons including business optimisation, of reaction to new business opportunities, of risk / threat mitigation, of operational emergencies, etc. It becomes therefore clear that a

well designed governance framework is a prerequisite to successfully implementing a SOA. The authors are involved in an activity at BT Innovate that is developing such a governance framework focusing on fulfilling security and dependability requirements in Service Oriented Infrastructures. More details on the objectives of such a SOA governance framework are given in [1].

### 3 Anatomy of Governance Framework

The presentation will be a live demonstration of composable and flexible security using connected systems.

The presenters have developed a SOA based infrastructure for security governance that meets the requirements and specifications introduced in [2][3][4].

In order to demonstrate the security governance, the authors have developed a security gateway that manages the security of web services that are exposed through it by the way of a security profile.

The security profile is defined by a taxonomy, presented in Figure 1, that describes the set of infrastructure services that are required for security (e.g. policy enforcement point, identity management, access control). This taxonomy is completed by sets of additional constraints such as policy templates; inter infrastructures coordination and management processes. Managed, these elements allow dynamically selecting and composing appropriate services to provide security and change it on the fly when necessary (e.g. detection of a security threat, change of requirements).

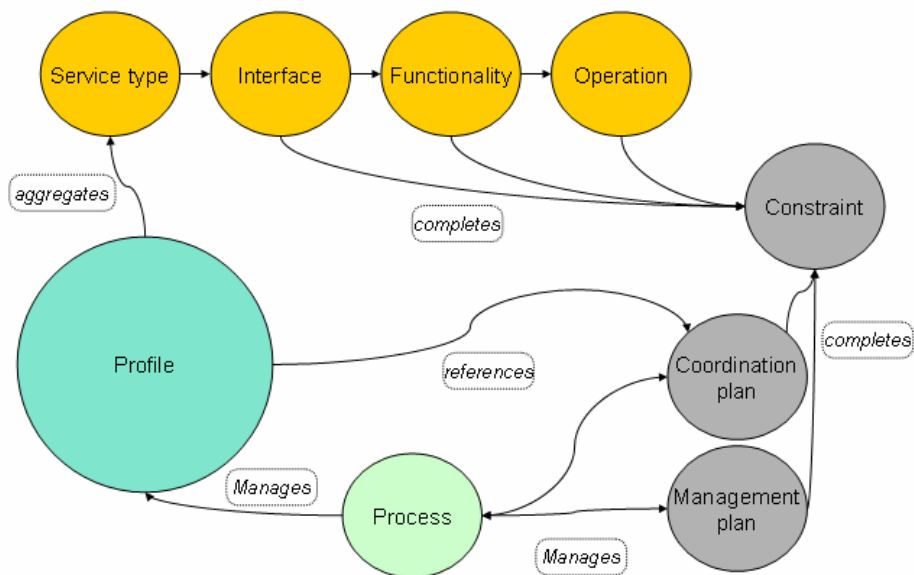


Fig. 1. Profile description taxonomy

## 4 Benefits and Unique Selling Points of the Solution

The innovations of the SOA based security governance demonstration is the increased usability it brings to security for non specialists that just want to expose their resources in a secured way, the semantically enhanced management capability to configure security at an abstract or more concrete level for security professionals, the improved visibility of consumed resources' security related properties it and finally the added flexibility to security management it allows. In addition, many different distributed systems such as services or mashups can make use of the security governance infrastructure presented.

One of the main innovations of the selected architecture is its modularity. Indeed, being based on a SOA makes possible to manage each module independently as is possible for Web services. In addition, this allows applying the same advantages provided to the supported SOA to the SOI-GGW itself. This concretely translates into a higher reliability and security in addition to the flexibility gain. Finally, using a SOA allows rendering the SOI-GGW extendable in case supplementary requirements should arise.

Another innovation is that this solution aims at addressing the issue of business capability exposure and management, unlike most other frameworks that only target the visibility, policy management and administration or resource life-cycle management.

An additional innovation stems from the fact that negotiation with different governance frameworks is thought of from the start. Where other governance solutions offer interoperability based upon the use of programmable API, shared interface exposed on the network or even for WSRR, the SOI-GGW proposed a policy and well defined schemes driven approach on the top of its SOA. These two elements put together offer a high interoperability from the semantic and technical points of view in both operational and managerial sides of the governance.

The final innovation brought by the security governance model adopted is that it allows managing many different types of services in various contexts. This is made possible by the SOA used to model it and the fact that more and more industries choose to deliver their offers as services. Good examples can be found in network services such as IPTV or IMS. The next two sections are geared towards presenting and demonstrating this innovation point.

## Acknowledgement

The authors would like to thank Dr. Theo Dimitrakos, Head of the Security Architecture Group at BT Innovate, for his extensive contribution in the SOA security governance project.

## References

- [1] Dimitrakos, T., Brossard, D., de Leusse, P.: Securing Business Operation in SOA. *BT Technology Journal* 27(2) (December 2008)
- [2] de Leusse, P., Periorellis, P., Watson, P., Maierhofer, A.: Secure & Rapid Composition of Infrastructure Services in the Cloud. In: *The Second International Conference on Sensor Technologies and Applications, SENSORCOMM 2008*, Cap Esterel, France, August 25-31, 2008. IEEE Computer Society, Los Alamitos (2008)

- [3] de Leusse, P., Periorellis, P., Watson, P., Dimitrakos, T.: A semi autonomic infrastructure to manage non functional properties of a service. In: UK e-Science All Hands Meeting 2008, Edinburgh, UK, September 8-11 (2008)
- [4] de Leusse, P., Periorellis, P., Dimitrakos, T., Watson, P.: An Architecture for Non Functional Properties Management in Distributed Computing. In: 3rd International Conference on Software and Data Technologies (ICSOFT 2008) (2008)

# Security and Trust Management for Virtual Organisations: GridTrust Approach

Syed Naqvi<sup>1</sup> and Paolo Mori<sup>2</sup>

<sup>1</sup> Centre d'Excellence en Technologies de l'Information et de la Communication (CETIC),  
29/3 Rue des Frres Wright, 6041 Charleroi, Belgium

syed.naqvi@cetic.be

<sup>2</sup> Istituto di Informatica e Telematica Consiglio Nazionale delle Ricerche (IIT-CNR),  
1 Via G. Moruzzi, 56124, Pisa, Italy

paolo.mori@iit.cnr.it

**Abstract.** The GridTrust Security Framework (GSF) offers security and trust management for the next generation Grids (NGG). It follows a vertical approach for Grid security from requirements level right down to application and middleware levels. New access control models for collaborative computing, such as the usage control model (UCON), are implemented for securing the Grid systems. The GSF is composed of security and trust services and tools provided at the middleware and Grid foundation middleware layers. GSF addresses three layers of the NGG architecture: the Grid application layer, the Grid service middleware layer, and the Grid foundation layer. The framework is composed of security and trust services and tools provided at the middleware and Grid foundation middleware layers. GSF provides policy-driven autonomic access control solutions that provide a continuous monitoring of the usage of resources by users.

## 1 Introduction

A secure Virtual Organisation (VO) requires that security objectives and requirements have been defined and are enforced throughout the VO lifecycle. GridTrust has employed a goal oriented requirements engineering method that is tailored for defining VO security objectives and refining them into enforceable security policies. An Eclipse-based policy design tool is developed that allows specifying and refining security objectives into requirements.

The usage control model (UCON) is a new access control paradigm proposed by Park and Sandhu [4] that encompasses and extends different existing models. Its main novelty, in addition to the unification view, is based on continuity of usage monitoring and mutability of attributes. This model is employed in the GridTrust project due to its suitability for managing access/usage control in Grid systems. GridTrust defines a policy specification language, POLPA, which is a process description language that is able to express the basic policy models defined by UCON.

GridTrust explored a utility-based model for reputation, which in contrast to most other reputation models that require direct feedback from users, builds the reputation from information provided by monitoring systems, making it suitable for Grids.

## 2 The GridTrust Project

The GridTrust framework (as depicted in Fig. 1) addresses three layers of the NGG architecture: the Grid application layer, the Grid service middleware layer, and the Grid foundation layer. The framework is composed of trust and security services and tools as indicated in the figure. The trust and security services are provided at the Grid service middleware and Grid foundation middleware layer. The services all use usage control policies. The services at the service middleware layer are the following: a Secure Resource Broker, a Reputation service, and a Service Level Usage Control Service. At the Grid foundation middleware layer fine grained continuous computational Usage Control service is provided.

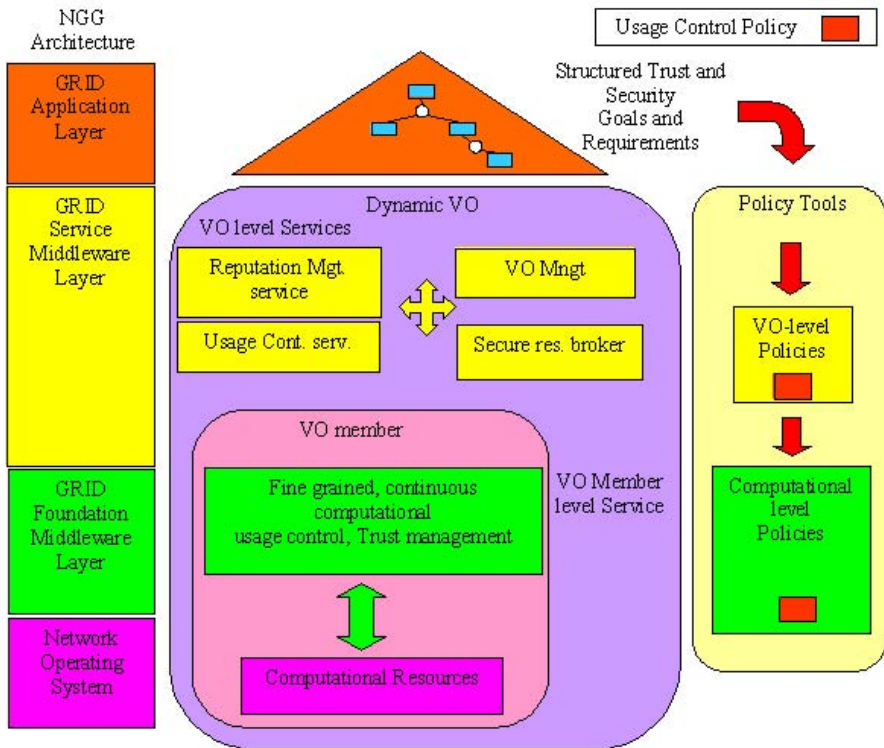


Fig. 1. GridTrust framework

### 2.1 Technical Approach

The Secure Resource Broker is invoked by the VO owner to retrieve the set of resources required to set up his VO. The VO owner also specifies the security requirements of the services he needs. The Reputation service keeps track of the past behaviour of VO users. This service is exploited by other GSF services that need the current reputation of a VO user to perform the decision process. The Service Level Usage Control Service

is a coarse grained authorization service that enforces XACML security policies to regulate the access to Grid services. The Computational Usage Control Service, instead, monitors the behaviour of the Java applications executed on computational services on behalf of remote VO users. This service is an implementation of the UCON framework proposed by Sandhu [4] adapted for the Grid necessities, and enforces a security policy written in POLPA, a process algebra based security policy language [2]. The service evaluates the security policy to decide whether each security relevant action performed by the application is permitted on the computational resource, and revokes actions in progress when the right does not hold anymore. To evaluate the security policy, the Computational Usage Control service interacts with the Reputation service, in order to get the current value of the user's reputation attribute. Moreover, this service also reports to the Reputation service a feedback about the user behaviour on the computational resource.

The GridTrust framework policy tools aim to produce the security and trust policies needed by the different services. At the application level a requirements tool helps analysts define security and trust goals and requirements, and produces high-level security and trust policies. A policy refinement tool takes the abstract security and trust policies as input and refines them into service and computational level usage control policies. These usage control policies are used by the different trust and security services.

## 2.2 Innovation

The innovation of the GridTrust project is its integrated framework that provides a set of services performing the main Grid interactions in a secure way. These tools allow the Grid participant to create and manage VOs, to select resource providers having certain security requirements, to manage users' mutable attributes such as the reputation, and to execute Java applications on behalf of remote Grid users in secure way, i.e. performing a fine-grained and continuous monitoring of computational services according to the UCON model. The cutting edge advantage of the GridTrust framework is that it consists of not only a simple authorization system but it provides a set of services that, exploiting the usage control model, enhances the security of the whole Grid lifecycle, starting from the VO formation, including the execution of Java applications on the VO computational services, until the VO dissolution.

## 3 Impact and Perspectives

The GridTrust project aims to enable companies to set up and operate dependable VOs that are secure and trusted. The demonstration will exhibit the various functioning of GridTrust tools for the security design and trust requirements of the VO. VOs will allow companies to provide and to access Grid resources to achieve common goals. VOs are also valuable in the larger context of Service Oriented Architectures to set up "virtual" markets and to support collaboration between different units of a corporation or between cooperating players in the same market [3].

In order to support rapid formation of VOs, we use the concept of Virtual Breeding Environment (VBE). A VBE can be defined as an association of organisations adhering



to common operating principles and infrastructure with the main objective of participating in potential VOs. We have adopted the view that organisations participating in a VO are selected from a VBE [11].

**Acknowledgements.** The research leading to the results presented in this paper has received funding from the European Union's sixth framework programme (FP6) Project GRIDTRUST under grant agreement number 033817.

## References

1. Blasi, L., Arenas, A., Aziz, B., Mori, P., Rovati, U., Crispo, B., Martinelli, F., Massonet, P.: A Secure Environment for Grid-Based Supply Chains. In: Cunningham, P., Cunningham, M. (eds.) Proc. eChallenges Conference 2008. Collaboration and the Knowledge Economy: Issues, Applications, Case Studies. IOS Press, Amsterdam (2008)
2. Martinelli, F., Mori, P.: A Model for Usage Control in GRID Systems. In: Proceeding of Grid-STP 2007, International Workshop on Security, Trust and Privacy in Grid Systems at SecureComm 2007. IEEE Computer Society, Los Alamitos (2007)
3. Naqvi, S., Massonet, P., Aziz, B., Arenas, A., Martinelli, F., Mori, P., Blasi, L., Cortese, G.: Fine-grained continuous usage control of service based Grids the GridTrust Approach. In: Mähönen, P., Pohl, K., Priol, T. (eds.) ServiceWave 2008. LNCS, vol. 5377, pp. 242–253. Springer, Heidelberg (2008)
4. Sandhu, R., Park, J.: The UCONABC Usage Control Model. ACM transaction on Information and System Security 7(1), 129–174 (2004)

# Common Capabilities for Trust and Security in Service Oriented Infrastructures

David Brossard<sup>1</sup> and Maurizio Colombo<sup>2</sup>

<sup>1</sup> BT Innovate, PP13D Orion Building, Adastral Park,  
IP5 3RE Martlesham Heath, England  
david.brossard@bt.com

<sup>2</sup> Istituto di Informatica e Telematica, Consiglio Nazionale delle Ricerche,  
via G. Moruzzi, 1, 56124 Pisa, Italy  
maurizio.colombo@iit.cnr.it

**Abstract.** In order to achieve agility of the enterprise and shorter concept-to-market timescales for new services, IT and communication providers and their customers increasingly use technologies and concepts which come together under the banner of the Service Oriented Infrastructure (SOI) approach. In this paper we focus on the challenges relating to SOI security. The solutions presented cover the following areas: i) identity federation, ii) distributed usage & access management, and iii) context-aware secure messaging, routing & transformation. We use a scenario from the collaborative engineering space to illustrate the challenges and the solutions.

## 1 Introduction

Today's enterprise is more pervasive with a mobile workforce, outsourced data centers, different customer engagements. This increases the need for securing end-to-end transactions between businesses and the customer. The presence of multiple authorities and complex relationships regarding the ownership of resources and information means that multiple administrators must be able to define policies about entitlements, resources, and access. Policies enforced at the same point may be defined by administrators from different organizations. We need to manage identities over enterprise domains to control them and manage information disclosure; securely exposing business services by enforcing exposure policies defining what can be invoked, by whom. This paper briefly introduces security capabilities in the context of a collaborative engineering scenario. For in-depth analysis of these capabilities and their business benefits, please refer to [1].

## 2 Context

This paper presents three security capabilities: (1) the identity broker (SOI-STs) – it acts as an identity broker for each enterprise and manages the correlation of identities and security attributes within a security domain; (2) the authorization service (SOI-AuthZ-PDP) – it implements a XACML engine and enables distributed Access Control; and (3) the security gateway (SOI-SMG) – it is a security gateway which protects

services, network traffic, and application data. These capabilities will be demonstrated in the following scenario: an aerospace company, Alpha Aerospace (AA), is engaged in developing fuel-efficient aircraft. AA is looking into optimizing its wing design to decrease fuel consumption. To achieve this, it will need a set of mathematical algorithms, High Performance Computing (HPC) resources, as well as secure storage sites where to maintain the results. Alpha turns to a third party collaboration manager, Epsilon, which will look up suitable partners. Eventually three partners are invited: Beta Algorithms offers computation algorithms; Gamma Computing offers HPC; and Delta Storage offers secure storage. The high-level interaction is as follows: (1) an Alpha designer locates an algorithm at Beta to process wing data with; (2) the designer pushes the raw data with the algorithm retrieved in (1) to Gamma Computing where a job is created with the appropriate QoS and level of security; (3) the calculation job eventually terminates and sends its output to the data store at Delta Storage; (4) the Alpha Aerospace designer can now use the data stored at Delta.

### 3 Technical Approach

In distributed environments we cannot authenticate users or services against one single identity store. Yet there is a need to share an enterprise's users' identities with another enterprise. In order to achieve this, we can use the SOI-STS developed at BT to translate and contextualize a user's/service's internal identity into a collaboration-wide virtual identity.

The SOI-STS implementation that we will demonstrate is a WS-Trust-based web service which issues and validates security tokens that can be used to sign and/or encrypt XML messages. The STS can broker user attributes that can be used in access control decisions. Lastly, the STS is also responsible for federation establishment between each provider's STS. In our scenario, Alpha Aerospace's STS will contain the list of users at Alpha that wish to take part in the collaboration. These users can have different attributes (e.g. 'Designer') depending on their role in the collaboration. These will then be checked against a PDP for access control.

Because there is no longer a single identity store and due to the distributed nature of SOA, using access control lists or enterprise-based hierarchical decisions to control access to resources is no longer sufficient. Role-based access control is also limited in an environment where we may not necessarily know a priori all the roles. There is a need for richer access control rules that can be authored by multiple administrators in different domains.

The SOI-AuthZ-PDP implements XACML: it can be used to express fine access control rules: who (subject) can do what (action) on which service (resource)?

The SOI-AuthZ-PDP responds to a request by locating a group of policies that apply to that request. Delegated policies are validated according to the XACML 3.0 administrative delegation model before use. The validation involves looking for root policies which authorize the delegated policies in accordance to the constraints defined in the administrative policies.

The SOI-SMG is a security application gateway / security appliance that securely exposes services on the basis of network traffic, message content, and application

data. It acts as a message interceptor, decorator, router and enforcer. It is also the integration node in an SOA deployment (see following subsection).

The SOI-SMG is policy-based. The policies, expressed in XML, allow for rich, highly adaptive scenarios. The policies to be executed are located based on contextual information and therefore the SOI-SMG can be used in several collaborations concurrently while maintaining clear message flow segregation.

The SOI-SMG can express rules that will encrypt parts of an XML (SOAP) message e.g. with different keys if the parts are aimed at different recipients. Messages can also be signed to ensure message integrity. The key benefit is the ability to express these rules with the high-level policy language which translates into a graphical language within the SOI-SMG's management interface. The SOI-SMG is highly adaptive and can be reconfigured at runtime with zero downtime to cater for new security requirements or changes in deployment.

We have also envisioned a new policy framework model which clearly separates concerns between enforcement actions (this specifies the enforcement state, the actions that are to be enacted, their execution conditions), interceptor actions (this contains mapping between each available enforcement action and the computational entity that executes this action), and capability exposure (this is used to publish additional conditions for interacting with a protected resource). This model enables richer scenarios, as separate parts of the policies can be modified independently.

## 4 Innovation

The STS's key innovations lie in its modular architecture: it becomes easy to plug in new connectors e.g. identity store. Circles of trust become manageable: one can define & revoke trust relationships between providers at any time. The tokens issuance / validation can be so on the basis of a context: the STS will use a different federation definition and the issuance / validation process may be different altogether. Compliance to standards eases integration with legacy applications, WSs, ESBs...

The PDP's foremost innovation is its implementation of the XACML 3.0 draft. This enables delegation and obligations. Delegation enables distributed Access Control and Authorization: policies can come from different administrative sources. The PDP also requires that the policies be signed. This ensures we can check the authenticity of the policies and run audits. Lastly, the PDP can enable contextualization and segregation of policy execution, enabling one PDP to act as separate PDPs by segregating policy sets based on contextual information.

The SOI-SMG, being an XML-driven security enforcement capability, is extremely flexible and can be used to implement a wide array of scenarios and security enforcement policies. Much like the STS and the PDP, the SOI-SMG can be contextualized: the execution of enforcement policies can be context-aware. In addition, the SOI-SMG enables security for Application networks: this means applications that are exposed as network-enabled services can securely integrate over the network. Through its content and context aware policies enforced at the enterprise boundary, the SMG enables deperimeterization: this increases the likelihood of more services being used and shared. Lastly, the SOI-SMG's key benefit is its extensible policy

framework: by using XML to express policies, it lets administrators define and reuse finely granular rules.

## 5 Business Impact

The solution allows context-based policy differentiation enabling multiple scenarios to be run concurrently. The solution can constrain, combine, and validate policies from multiple authorities, breaking down monolithic views. The same policies can be validated in order to assess the correctness of the security enforced. The ability to audit those policies also enables compliance. This has become critical in a world teeming with laws and directives. Other benefits include the reduction of integration timescales of value-adding services. The SOI-SMG acts as an integration node: it encourages the reuse of common infrastructure thus reducing costs. Lastly, seamless service interaction within and outside corporate boundaries implies end-users will not feel the difference between home or remote services.

**Acknowledgments.** The paper is partly the result of work from FP6 European projects TrustCoM and BEinGRID. It has also been fuelled by The SOA Architectures team at BT Innovate.

## References

1. Dimitrakos, T., et al.: Securing Business Operations in an SOA. *BT Technology Journal* 27(1) (April 2009)
2. Brossard, D., Prieto Martinez, J.L.: A Virtual Hosting Environment for Distributed Online Gaming. In: Ferrari, E., Li, N., Bertino, E., Karabulut, Y. (eds.) *TM 2009, IFIP AICT*, vol. 300, pp. 314–317. Springer, Heidelberg (2009)

# A Virtual Hosting Environment for Distributed Online Gaming

David Brossard and Juan Luis Prieto Martinez

BT Innovate, PP13D Orion Building, Adastral Park,  
IP5 3RE Martlesham Heath, England  
{david.brossard,juanluis.prietomartinez}@bt.com

**Abstract.** With enterprise boundaries becoming fuzzier, it's become clear that businesses need to share resources, expose services, and interact in many different ways. In order to achieve such a distribution in a dynamic, flexible, and secure way, we have designed and implemented a virtual hosting environment (VHE) which aims at integrating business services across enterprise boundaries and virtualising the ICT environment within which these services operate in order to exploit economies of scale for the businesses as well as achieve shorter concept-to-market time scales. To illustrate the relevance of the VHE, we have applied it to the online gaming world. Online gaming is an early adopter of distributed computing and more than 30% of gaming developer companies, being aware of the shift, are focusing on developing high performance platforms for the new online trend.

## 1 Introduction

Internet-based gaming offers challenging features such as interactivity with multiple players, latency requirements, high-performance servers for game execution, richness of the games, billing requirements, user privacy issues. But running a gaming infrastructure can be costly especially if under-used. It is critical that online gaming industries adopt technology that let them be more flexible and adaptive so they can securely address their customers' needs while delivering with the minimal amount of resources the quality the customer has requested. To address these issues, we have developed a Virtual Hosting Environment (VHE): it offers an advanced ICT environment that enables integrating business services to operate across enterprise boundaries over a virtualized ICT infrastructure. New reusable capabilities can be exposed as network-hosted services and be seamlessly introduced in the VHE to enhance its generic functionality or meet market-specific needs. This paper will describe the VHE applied to an online gaming scenario.

## 2 Context

The online gaming scenario contains four main actors: (1) the game player (GP) is the end consumer and is looking for a particular game title and game plan which suits his means; (2) the Online Game Platform Provider (OLG) identifies business opportunities,

selects the game titles to offer, looks for game providers, forms the virtual organization (VO), and manages the match lifecycle for the selected game title; (3) the Games Server Provider (GA) offers game host environments, advertises game application services that meet certain QoS levels, and participates in VOs formed by OLG; (4) the Infrastructure Provider (ISP) enables VO lifecycle management, mediates GA discovery by the OLG, brokers agreements, facilitates the use of common capabilities (CC) offered by 3<sup>rd</sup> party providers, and offers business service registries. Such CC include security capabilities (see [1,2]).

There are four key areas in the gaming scenario: (1) the virtual organization management service (VOMS) is responsible for the definition & establishment of secure federations between different partners. It maintains and publishes partners, services, policies, and roles. It drives partner selection based on service-level requirements; (2) the B2B gateway runs at each partner's site and interfaces between the VOMS and that partner's resources. It maintains its policies, list of services, and users. It enables resource virtualization and interfaces with the hosting environments (HE) as well as the value-added services (VAS); (3) the HE is a set of physical servers or virtual machines that can host the application to be virtualized. It offers interfaces to load applications, create instances, and capture QoS measurements; (4) VAS live in the cloud and can be used to support key areas such as security [1,2]. Other VAS include presence or telephony services e.g. VoIP.

Andago, a games company, wants to offer new games. It is looking for more games and hosting resources. It wants to delegate the hosting and other non-functional requirements to 3<sup>rd</sup> parties. Andago wants to focus on their game portal and their users. Hence, Andago requests a VO creation: it will invite partners that will provide hosting as well as security, auditing, and SLA services. Andago invites suitable partners via the VOMS. Sunny and Saygah, two hosting enterprises will fulfill the GS role. Andago will fulfill the OLG role.

### 3 Technical Approach

Using the VOMS, Andago associates each role with policies which are sent to each partner's B2B gateway where they are refined and stored. Here, we decide to send access control policies for the GS and claims for the OLG. They will be used to configure the relevant infrastructure services. Once each partner accepts the VO invitation, Andago initiates the VO creation and orders are sent to each partner's B2B gateway to configure their infrastructure and to make their services available in the newly created VO. Each B2B gateway then looks for a suitable infrastructure profile to use to create its own view of the VO. Profiles point to different infrastructure services such as the security services in [2] or SLA services. In this scenario, we make use of a security token service (STS), an XML security gateway, and an authorization service. The XML security gateway is used to virtualize, contextualize, and expose each partner's selected service(s). At this point, each partner's B2B gateway governance layer creates a new federation at the corresponding STS and pushes it the partner business cards. The global UDDI is updated: each partner's services are made available for virtualization within the VO. At Sunny and Saygah, the advertised services are exposed – in this case the management web service for the hosted game (called a watcher). The infrastructure services are then configured with the relevant policies.

Andago then consumes the services it wishes from Sunny and Saygah: Andago is after the EnemyTerritory game hosted at Sunny & Saygah.

For the user, response time, availability, and gamer account security are critical. The player's interaction starts when he goes to Andago's website. He logs into Andago's web portal which manages users, games, and online communities. The user then chooses a game to play. Up to this point, all interactions take part solely between Andago and the end user. Once the player has selected a game, he chooses a match to join. This is where the VHE kicks in. A match is in fact a given specific VO with a virtualized exposed web service for one of the game servers (GS) selected e.g. Sunny. Once the player has selected a match, Andago starts the virtualization of its own client application, Agasy, which liaises between its platform and Sunny's virtualized watcher. At this stage, Andago is virtualising and exposing an instance of Agasy. The latter exchanges management & monitoring messages with the Sunny watcher instance. When it calls the watcher instance, the request goes through the supporting infrastructure previously configured by the B2B gateway. The demonstrator will focus on the details of the interactions and how message exchanges are secured. In the following paragraph we have a closer look at which particular capabilities are used and for which purposes.

The key capabilities are the VOMS, the Federation Manager & Identity Broker (STS), the Authorization Service (PDP), the XML Enforcement Gateway (PEP), and the SLA monitor (SLAm). The security infrastructure comes into play as follows: outbound, the message is intercepted by Andago's XML security gateway. There, the message is (1) checked and decorated with a token issued by Andago's STS; (2) checked for compliance against Andago's authorization rules at the PDP; (3) encrypted with a proof-of-possession (PoP) key embedded in the SAML token; and (4) sent to the virtual endpoint of the watcher instance at Sunny. Inbound, at Sunny's gateway, (1) the SAML token is extracted and sent to the STS; it (2) extracts the PoP and Agasy's identity claims; (3) the XML gateway decrypts the message and requests an authorization decision from the PDP based on the extracted claims; (4) the PDP returns a decision based on the claim, target, action, and environment information; (5) the gateway then forwards the request to the watcher instance. The demonstrator will also illustrate the use of an SLA system.

## 4 Innovation

Overall, the VHE drives and achieves total separation of concerns between functional and non-functional requirements. This in turn allows: (1) agile composition of services: offering capabilities as services is one step towards 'SaaS'; (2) optimized resource usage and reusable infrastructure; (3) increased flexibility: the VHE enables a total management & adaptation of exposed services with zero downtime; (4) enriched user experience: with the SLA framework of the VHE in particular. Each capability has specific innovative features discussed in [1,2].

## 5 Business Impact

We estimate up to 80% of enterprises anticipate the following benefits from the VHE: loosely coupled systems, service reuse, composite applications built by combining



services, enabling response to changing market requirements and first-to-market competitive advantage, optimization of end-to-end processes, enabling a higher degree of automation, and compliance mechanisms. The VHE enables the customer to adopt a low-risk approach to SOA and increased return on investment. It provides common technologies that enable business processes to be added, changed, and removed easily. The VHE can support non-functional requirements e.g. QoS-based service publication, process-driven service composition; federated identity & access control, and secure messaging.

The core market relevant to VHE is a segment of the Infrastructure Management services market. The opportunities for Communication and IT services companies in this market in Europe are predicted to reach € 61.3 billion by 2009. Services empower a wide variety of customers in multiple domains, e.g. Defence, Finance, or Media. The VHE offers a strategic advantage to Communications and IT Services companies when competing with SOA vendors. They increase their cost effectiveness, decrease concept-to-market timescales and create economies of scale.

**Acknowledgments.** The paper is the result of work from EU projects TrustCoM, BE-inGRID, and GridTrust. In addition, components from EU project GRASP are used in the demonstration.

## References

1. Dimitrakos, T., et al.: Securing Business Operations in an SOA. *BT Technology Journal* 27(1) (April 2009)
2. Brossard, D., Colombo, M.: Common Capabilities for Trust & Security in Service Oriented Infrastructures. In: Ferrari, E., Li, N., Bertino, E., Karabulut, Y. (eds.) *TM 2009, IFIP AICT*, vol. 300, pp. 310–313. Springer, Heidelberg (2009)

# Author Index

- Ahmed, Mohamed 47  
Andrulis, Jonas 267  
Arenas, Alvaro 99  
Atalay, Nart Bedin 15  
Aziz, Benjamin 99  
  
Bertino, Elisa 65, 164  
Bicakci, Kemal 15  
Brossard, David 302, 310, 314  
Brunie, Lionel 65  
  
Camp, L. Jean 234  
Chumash, Tzvi 117  
Cohen, Robin 252  
Colombo, Maurizio 310  
  
de Leusse, Pierre 302  
  
Erdeniz, Burak 15  
  
Gupta, Anubha 196  
Gurbaslar, Hakan 15  
  
Hailes, Stephen 47  
Haller, Jochen 267  
Hasan, Omar 65  
Helvik, Bjarne E. 283  
Herrmann, Peter 133, 149  
  
Jensen, Christian Damsgaard 81, 298  
  
Karabulut, Yuceel 267  
Kerr, Reid 252  
Khan, Latifur 196  
Knapskog, Svein J. 133, 283  
Korsgaard, Thomas Rune 298  
  
Lee, Adam J. 176  
Lefèvre, Thomas 298  
  
Martinelli, Fabio 99  
Moe, Marie E.G. 283  
Mori, Paolo 306  
  
Naqvi, Syed 306  
Nielek, Radoslaw 1  
  
Öztürk, Pinar 149  
  
Perano, Kenneth J. 176  
Pierson, Jean-Marc 65  
Pitsilis, Georgios 30  
Prieto Martinez, Juan Luis 314  
  
Ramanujam, Sunitha 196  
Refsdal, Atle 215  
  
Sandhu, Ravi 164  
Seida, Steven 196  
Stagni, Federico 99  
Stølen, Ketil 215  
  
Tavakolifard, Mozhgan 133, 149  
Thuraisingham, Bhavani 196  
  
Viecco, Camilo H. 234  
  
Wawer, Aleksander 1  
Weinhardt, Christof 267  
Wierzbicki, Adam 1  
Winslett, Marianne 176  
  
Xu, Shouhuai 164  
  
Yao, Danfeng 117  
Yuceel, Mustafa 15