Joaquin Garcia-Alfaro
Guillermo Navarro-Arribas
Ana Cavalli
Jean Leneutre (Eds.)

LNCS 6514

Data Privacy Management and Autonomous Spontaneous Security

5th International Workshop, DPM 2010 and
3rd International Workshop, SETOP 2010
Athens, Greece, September 2010, Revised Selected Papers

Springer

Lecture Notes in Computer Science 6514

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Joaquin Garcia-Alfaro
Guillermo Navarro-Arribas
Ana Cavalli
Jean Leneutre (Eds.)

Data Privacy Management and Autonomous Spontaneous Security

5th International Workshop, DPM 2010 and
3rd International Workshop, SETOP 2010
Athens, Greece, September 23, 2010
Revised Selected Papers

Springer

Volume Editors

Joaquin Garcia-Alfaro
IT/TELECOM Bretagne, Campus de Rennes
2 Rue de la Châtaigneraie, 35512 Cesson Sévigné, Cedex, France
E-mail: joaquin.garcia@telecom-bretagne.eu

Guillermo Navarro-Arribas
IIIA-CSIC, Campus UAB
08193 Bellaterra, Spain
E-mail: guille@iiia.csic.es

Ana Cavalli
IT/TELECOM SudParis
9 Rue Charles Fourier, 91011 Evry Cedex, France
E-mail: ana.cavalli@it-sudparis.eu

Jean Leneutre
IT/TELECOM ParisTech
46 Rue Barrault, 75634 Paris Cedex 13, France
E-mail: jean.leneutre@telecom-paristech.fr

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-19347-7 e-ISBN 978-3-642-19348-4
DOI 10.1007/978-3-642-19348-4
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2011921004

CR Subject Classification (1998): K.6.5, E.3, K.4.1, K.4.4, C.2, C.3, D.4.6, H.3.5

LNCS Sublibrary: SL 4 – Security and Cryptology

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

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Foreword from the Program Chairs of DPM 2010

This volume contains the proceedings of the 5th Data Privacy Management International Workshop (DPM 2010). It includes a revised version of the papers selected for presentation at the workshop. The aim of DPM is to promote and stimulate international collaboration and research exchange on novel data privacy topics.

Organizations are increasingly concerned about the privacy of information that they manage (several people have filed lawsuits against organizations violating the privacy of customers' data). Thus, the management of privacy-sensitive information is very critical and important for every organization. This poses several challenging problems, such as how to translate the high-level business goals into system-level privacy policies, administration of privacy-sensitive data, privacy data integration and engineering, privacy access control mechanisms, information-oriented security, and query execution on privacy-sensitive data for partial answers.

This 5th edition of the Data Privacy Management International Workshop presented nine regular papers and three keynotes talks. It was co-located with the ESORICS 2010 conference in Athens, Greece, and took place on September 23, 2010. Previous issues of the DPM workshop were: 2009 Saint Malo (France), 2007 Istanbul (Turkey), 2006 Atlanta (USA), and 2005 Tokyo (Japan).

We would like to express our sincere appreciation for all the support we received from the General Chairs of DPM 2010, Frédéric Cuppens and Nora Cuppens-Boulahia. Thanks very much for your contribution to the success of the event. Our special thanks also go to our distinguished keynote speakers, Gene Tsudik (University of California, Irvine), Vicenç Torra (Artificial Intelligence Research Institute, Spanish National Research Council), and Sabrina De Capitani di Vimercati (University of Milan), for accepting our invitation and for their presence during the event and talks.

We also express our gratitude to the ESORICS 2010 organization team. Thank you Sokratis Katsikas, Dimitris Gritzalis, Pierangela Samarati, Nikolaos Kyrloglou, and Marianthi Theoharidou, for all your help with the local arrangements of DPM. Many thanks go to the DPM 2010 Program Committee members and the external reviewers, for their help, availability and commitment. Last but by no means least we thank all the authors who submitted papers and all the workshop attendees.

Finally, we want to acknowledge the support received from the sponsors of the workshop: the UNESCO Chair in Data Privacy; projects ARES-CONSOLIDER

INGENIO 2010 CSD2007-00004 and eAEGIS TSI2007-65406-C03-02/TSI2007-65406-C03-01 from the Spanish MICINN; the Artificial Intelligence Research Institute (IIIA-CSIC); the Internet Interdisciplinary Institute (IN3) from the Universitat Oberta de Catalunya; Telecom Bretagne (Institut Telecom); and the Spanish section of the IEEE.

September 2010

Joaquin Garcia-Alfaro
Guillermo Navarro-Arribas

Foreword from the Program Chairs of SETOP 2010

The International Workshop on Autonomous and Spontaneous Security (SETOP), co-located with the ESORICS symposium, presents research results on all aspects related to spontaneous and autonomous security. This year, the third issue of SETOP was held in Athens on September 23, 2010.

With the need for evolution, if not revolution, of current network architectures and the Internet, autonomous and spontaneous management will be a key feature of future networks and information systems. In this context, security is an essential property. It must be considered at the early stage of conception of these systems and designed to be also autonomous and spontaneous. Future networks and systems must be able to automatically configure themselves with respect to their security policies. The security policy specification must be dynamic and adapt itself to the changing environment. Those networks and systems should interoperate securely when their respective security policies are heterogeneous and possibly conflicting. They must be able to autonomously evaluate the impact of an intrusion in order to spontaneously select the appropriate and relevant response when a given intrusion is detected.

Autonomous and spontaneous security is a major requirement of future networks and systems. Of course, it is crucial to address this issue in different wireless and mobile technologies available today such as RFID, Wifi, Wimax, 3G, etc. Other technologies such as ad hoc and sensor networks, which introduce new type of services, also share similar requirements for an autonomous and spontaneous management of security.

The accepted papers addressed several specific aspects of the previously cited topics, as for instance the autonomic administration of security policies, secure P2P storage, RFID authentication, anonymity in reputation systems, etc. The high quality of the SETOP 2010 communications facilitated a stimulating exchange of ideas between the participants of the workshop. These proceedings contain the revised versions of the accepted papers. SETOP 2010 was honored to have three distinguished keynote speakers: Gene Tsudik, from the University of California, Irvine (UCI); Vicenç Torra, from the Artificial Intelligence Research Institute of the Spanish National Research Council (IIIA-CSIC); and Sabrina De Capitani di Vimercati, from the University of Milan. Thank you, Gene Tsudik, Vicenç Torra, and Sabrina De Capitani di Vimercati, for having accepted our invitation.

We would like to thank the General Chair of SETOP 2010, Frédéric Cuppens, and the General Chair of ESORICS, Sokratis Katsikas. The Organizing

Committee from ESORICS 2010 (N. Kyrloglou and M. Theoharidou) together with the SETOP 2010 Organizing Chairs (N. Oualha and W. Mallouli) helped with the local organization. We also thank all the SETOP 2010 Program Committee members for their help, availability and commitment. We also extend our thanks to Nora Cuppens-Boulahia.

September 2010

Ana Cavalli
Jean Leneutre

5th International Workshop on Data Privacy Management — DPM 2010

Program Committee Chairs

Joaquin Garcia-Alfaro	TELECOM Bretagne, France
Guillermo Navarro-Arribas	IIIA-CSIC, Spain

Workshop General Chairs

Frédéric Cuppens	TELECOM Bretagne, France
Nora Cuppens-Boulahia	TELECOM Bretagne, France

Program Committee

Alessandro Acquisti	Carnegie Mellon University, USA
Mohd Anwar	University of Calgary, Canada
Michel Barbeau	Carleton University, Canada
Elisa Bertino	Purdue University, USA
Marina Blanton	University of Notre Dame, USA
Nikita Borisov	University of Illinois, USA
Joan Borrell	Autonomous University of Barcelona, Spain
Milan Bradonjic	Los Alamos National Laboratory, USA
Jordi Cabot	AtlanMod, INRIA, France
Iliano Cervesato	Carnegie Mellon University, Qatar
Valentina Ciriani	University of Milan, Italy
Frédéric Cuppens	TELECOM Bretagne, France
Nora Cuppens-Boulahia	TELECOM Bretagne, France
Ernesto Damiani	University of Milan, Italy
Josep Domingo-Ferrer	Rovira i Virgili University, Spain
David Evans	University of Cambridge, UK
Simon Foley	University College Cork, Ireland
Philip W.L. Fong	University of Calgary, Canada
Joaquin Garcia-Alfaro	TELECOM Bretagne, France
Stefanos Gritzalis	University of the Aegean, Greece
Jordi Herrera	Autonomous University of Barcelona, Spain
Wei Jiang	Missouri University, USA
Evangelos Kranakis	Carleton University, Canada
Javier Lopez	University of Malaga, Spain
Fabio Massacci	Università di Trento, Italy
Guillermo Navarro-Arribas	IIIA-CSIC, Spain
Andreas Pashalidis	K.U. Leuven, Belgium

Tomas Sander	Hewlett-Packard Labs, USA
Thierry Sans	Carnegie Mellon University, Qatar
Vicenç Torra	IIIA-CSIC, Spain
Nicola Zannone	Eindhoven University of Technology, The Netherlands

Organizing Committee

Frédéric Cuppens	TELECOM Bretagne, France
Nora Cuppens-Boulahia	TELECOM Bretagne, France
Joaquin Garcia-Alfaro	TELECOM Bretagne, France
Guillermo Navarro-Arribas	IIIA-CSIC, Spain

External Referees

Spyros Kokolakis	University of the Aegean, Greece
Payman Mohassel	University of Calgary, Canada
Bo Qin	Rovira i Virgili University, Spain
Evangelos Rekleitis	University of the Aegean, Greece
Daniel Trivellato	Eindhoven University of Technology, The Netherlands
Lei Zhang	Rovira i Virgili University, Spain

3rd International Workshop on Autonomous and Spontaneous Security — SETOP 2010

Program Committee Chairs

Ana Cavalli	TELECOM SudParis, France
Jean Leneutre	TELECOM ParisTech, France

General Chair

Frédéric Cuppens	TELECOM Bretagne, France

Organizing Committee Chairs

Wissam Mallouli	Montimage, France
Nouha Oualha	TELECOM ParisTech, France

Program Committee

Michel Barbeau	Carleton University, Canada
Christophe Bidan	Supélec, France
Ana Cavalli	TELECOM SudParis, France
Hakima Chaouchi	TELECOM SudParis, France
Claude Chaudet	TELECOM ParisTech, France
Yves Correc	DGA/CELAR, France
Frédéric Cuppens	TELECOM Bretagne, France
Hervé Debar	France Télécom R&D, France
Jose M. Fernandez	Ecole Polytechnique de Montréal, Canada
Noria Foukia	University of Otago, New Zealand
Alban gabillon	University of Polynesie Française, France
Joaquin Garcia-Alfaro	TELECOM Bretagne, France
Roland Groz	LIG, France
Evangelos Kranakis	Carleton University, Canada
Marc Lacoste	Orange Labs, France
Jean Leneutre	TELECOM ParisTech, France
Javiez Lopez	University of Malaga, Spain
Maryline Maknavicius	TELECOM SudParis, France
Wissam Mallouli	Montimage, France
Amel Mammar	TELECOM SudParis, France
Catherine Meadows	Naval Research Laboratory, USA
Refik Molva	EURECOM, France
Nouha Oualha	TELECOM ParisTech, France

Radha Poovendran	University of Washington, USA
Yves Roudier	Eurecom, France
Juan Carlos Ruiz	UPV, Spain
Thierry Sans	Carnegie Mellon University, Qatar
Bachar Wehbi	Montimage, France

Steering Committee

Ana Cavalli	TELECOM SudParis, France
Nora Cuppens-Boulahia	TELECOM Bretagne, France
Frédéric Cuppens	TELECOM Bretagne, France
Jean Leneutre	TELECOM ParisTech, France
Yves Roudier	Eurecom, France

Table of Contents

Keynote Talks

Data Privacy Management

Autonomous and Spontaneous Security

Towards Knowledge Intensive Data Privacy

Vicenç Torra

IIIA, Institut d'Investigació en Intel·ligència Artificial
CSIC, Consejo Superior de Investigaciones Científicas
Campus UAB s/n, 08193 Bellaterra,
Catalonia, Spain
vtorra@iiia.csic.es

Abstract. Privacy preserving data mining tools only use in a limited way information and knowledge other than the data base being protected. In this paper we plead on the need of knowledge intensive tools in data privacy. More especifically, we discuss the role of knowledge related tools in data protection and in disclosure risk assessment.

1 Introduction

Privacy Preserving Data Mining (PPDM) [25,30] and Statistical Disclosure Control (SDC) [31] are two related fields with a similar goal: the protection of data so that no confidential information can be disclosed. Differences between these two fields root on their origins. SDC, with its origin in National Statistical Offices, is more focused on tools that permit data users to obtain appropriate results on statistical typical analysis. In contrast, PPDM rooted in the Data Mining and Machine Learning communities, focus on tools for classification.

The data-driven approach (perturbative approach) is one of the approaches developed in both areas for data protection. It consists of modifying the original data before transferring them to third parties. This modification is to ensure some levels of privacy. Nevertheless, current tools have important restrictions on the type of data being protected and almost no knowledge is used neither in the protection process nor in the evaluation of risk.

In this paper, we plead for the use of knowledge in data privacy. We will discuss about the development of methods and tools to exploit the knowledge in the data to be protected, and how this knowledge can be used effectively in the data protection process. Also, we discuss about the use of knowledge in disclosure risk assessment. We will present some of our results in this area.

In other words, we will discuss (i) the need, and some preliminary work towards, knowledge intensive data protection methods, and (ii) data-rich tools for evaluating the disclosure risk of the data to be released.

The structure of this paper is as follows. In Section 2 we review the role of knowledge in data privacy and in Section 3 we present some examples of the use of knowledge in data privacy. The paper finishes with some some lines of future work.

J. Garcia-Alfaro et al. (Eds.): DPM 2010 and SETOP 2010, LNCS 6514, pp. 1–7, 2011.

2 The Role of Knowledge in Data Privacy

Existing methods and measures for data privacy use only a limited amount of information. Protection methods usually restrict all the information they use to the data being processed. That is, the file to be protected. In a similar way, some measures as e.g. disclosure risk measures are only based on the data file to be protected and the protected file.

A few methods exist which are based on some additional information or knowledge. That is, for example, the case of some of the methods for achieving k-anonymity for categorical data. Such methods require a description of the terms of the variables as well as a classification of these terms in terms of a dendrogram. For some of the methods, this dendrogram is assumed to be known.

Nevertheless, tools in data privacy need to consider information and knowledge in a larger extend. Privacy and data quality is a function of the context, and, thus, they depend on the additional information available to intruders and data analyzers. Therefore, files need to be considered in such semantic context, and both data protection methods as well as disclosure risk measures should take this context into account.

Due to this, knowledge intensive tools for both risk assessment and data protection have to be developed. We outline the need of knowledge in these two settings.

Knowledge in data protection. Research is needed towards the use of an explicit representation of all knowledge related to the data (in particular, about the schema of the database, and constraints between attributes) and also of their privacy requirements. This will result into knowledge intensive data protection methods.

Knowledge and semantic context includes the fact that the protected data should satisfy on the one hand the data models, and on the other it should permit meaningful analysis. With respect to the former, just consider a protected data with negative ages, and with respect to the later, a protected data set with *random generalization* of ZIP codes or cities.

Categorical data and terms often have more *semantic deepth* than numerical data. Protection mechanisms should take this semantics under consideration.

Knowledge in risk assessment. Risk should be evaluated in a more accurate way. That is, it should be estimated taking into account not only the information that an intruder might possess (e.g., related databases) but also the information available in the web. New reidentification algorithms should be developed for exploiting all available information when assessing risk. Technologies as schema matching, ontology matching, and other semantic technologies should be taken into account, otherwise the corresponding risk is not detected. This will result into knowledge-rich reidentification algorithms.

Knowledge intensive data protection methods will improve largely the quality of the protected data, extending their application domain and simplifying its use; knowledge-rich risk assessment, taking into account the web and semantic technologies, will evaluate risk much better than current approaches.

In the next section, some particular examples following the two lines summarized above are described.

3 Using Knowledge in Data Privacy: Some Examples

In this section we review three different research problems in which we have contributed where knowledge has a role in data privacy. Two of them use knowledge for data protection, and the other one uses knowledge for risk assessment.

3.1 Constrained Data

Data bases are defined with constraints on the variables. These constraints, which can be either implicit or explicit, limit the possible values of the variables. For example, age is positive, the variable number of pregnacies should be settled to zero for a record when the record corresponds to a man, and total income is the sum of basic salary plus incentives.

Data protection should take these constraints into consideration. Nevertheless, it is usual that data driven (perturbative) methods do not take such constraints into account. In this case, the protected file might be inconsistent with the constraints even in the case that the original file satisfies all the constraints. For example, additive noise to ages might lead to negative values for some of the records.

A standard approach used by statistical offices is to protect the data and then apply a *data editing* tool to correct the errors.

In fact, data editing tools were defined in the statistical disclosure control community to make consistent the data obtained from e.g. surveys with the data model. I.e., data is modified so that no negative age is present in a data file. See e.g. [7,17] for details on data edit.

Actually, in real practice, data editing is one of the first processes applied to the data, even prior to any data protection method, so that any subsequent process is applied to consistent data. Taking this into account, data editing is applied twice when a data protection method is applied to the data. One is applied to the raw data to make it consistent with the data model, and the other just after the protection process.

However besides of the inconvenience of applying twice the data editing tools, the application of a these tools after the application of a data protection method may imply that the final protected data set looses some of the properties of the dataset. E.g., adding white noise to the data does not change the mean of the data. Nevertheless, if data consists of ages, the edition of the ages to remove negative values might imply that the mean of the file is modified.

Due to this, data protection methods need to be developed so that constraints on the data are used within the protection process. This is not the case for most of the existing data protection methods.

Some preliminary research exists [20,24] about developing data protection methods compliant with the constraints, based on an explicit representation of

them. We have contributed on the analysis of microaggregation in this setting, establishing the valid operations to construct the centroids or representative of the clusters [24] and we have developed a tool for automating the process. This tool [4] uses XML to represent the database and Schematron rules to represent our knowledge on the constraints of the database. Using this information, the tool selects the appropriate microaggregation algorithms for data protection. Our approach takes into account the following types of constraints:

- Constraints on the possible values.
- One variable governs the possible values of another one.
- Linear constraints.
- Non-linear constraints (focusing on multiplicative constraints).

See the references above for details.

3.2 Semantic Data Protection

Data to be protected against privacy leakage have some semantic context. This is especially so when they are categorical or correspond to terms in natural language. In this case meaning needs to be considered in the data protection context. This need is a must for linguistic terms and other semantical entities.

At present, some data protection tools of categorical data for achieving k-anonymity [18,19,22] already use explicit representation of the similarities of the terms in terms of dendrograms (hierarchical representations). For example, geographical information as ZIP codes, towns, counties and states can be represented hierarchically and then used to perform generalizations of the records to achieve k-anonymity of these records. Protection using these dendrograms can be seen as a semantic-based data protection. Nevertheless, the amount of knowledge used is rather limited. Incognito [9] and the hash-based algorithm in [21] are two examples of such systems.

Other tools [2,3,10,11] have been developed for e.g. microaggregation in which the terms are aggregated taking into account its meaning according to some ontology (e.g. Wordnet [12] and the Open Directory Project [16]).

For example, [3] focuses on the protection of text documents. It uses WordNet to protect sensitive terms in a document. More especifically, the goal is to produce an index of document terms where sensitive terms are replaced by non-sensitive ones. This is achieved through microaggregation of the terms in the documents and where cluster centers are vectors of non-sensitive terms obtained through the relations between the terms in wordnet.

[6] extends the microaggregation of query logs in [13] introducing a semantic approach when computing the similarities between queries. In this case, the semantic approach is based on the Open Directory Project (ODP).

3.3 Knowledge-Rich Disclosure Risk Assessment

Record linkage [28,32] is a versatile tool for measuring disclosure risk, as it can be applied to a large variety of scenarios including the case of publishing synthetic

data [27]. The rationale of using record linkage for risk assessment is that the goal of an intruder is to link her information with the protected dataset to extract new information. That is why, record linkage is used to model the attack linking the protected data set and another file that corresponds to the information of the intruder.

Current approaches for measuring risk using record linkage rely only on the published data and a file with a similar structure assumed to contain the information known by the intruder. Such intruder's file contains records represented using the same schema used for the protected file.

New approaches have to be considered so that record linkage can be applied to new attacks and scenarios. Data environment knowledge needs to be incorporated in the record linkage process. In addition, tools to learn or extract knowledge from the record linkage process are also needed.

Preliminary results in this area include, record linkage appliable when the intruder's file has a schema different than the one of the original file (see e.g. [23,29]), record linkage taking into account how data has been protected (see e.g. [14,15]), supervised record linkage to exploit some knowledge about the record linkage process, and parameter determination for record linkage (see e.g. [1]).

4 Conclusions

In this paper we have plead for the use of knowledge in data privacy. We have discussed three examples where this use is of interest. Two focusing on data protection and a third one focusing on disclosure risk assessment.

Nevertheless, these existing approaches use only a limited amount of information and knowledge, and further work and research needs to be performed to obtain real knowledge intensive tools in data privacy. The use of knowledge intensive tools will be beneficial in the whole process of data protection and dissemination.

Acknowledgements

Partial support by the Spanish MEC (projects ARES – CONSOLIDER INGENIO 2010 CSD2007-00004 – and eAEGIS – TSI2007-65406-C03-02) is acknowledged.

References

1. Abril, D., Navarro-Arribas, G., Torra, V.: Choquet Integral for Record Linkage (2010) (manuscript)
2. Abril, D., Navarro-Arribas, G., Torra, V.: Towards privacy preserving information retrieval through semantic microaggregation. In: Proc. 2010 IEEE/WIC/ACM Int. Conf. on Web Intelligence / Intelligent Agent Technology (WI/IAT 2010), Toronto, Canada (2010) (in press)

3. Abril, D., Navarro-Arribas, G., Torra, V.: Towards semantic microaggregation of categorical data for confidential documents. In: Torra, V., Narukawa, Y., Daumas, M. (eds.) MDAI 2010. LNCS (LNAI), vol. 6408, pp. 266–276. Springer, Heidelberg (2010)
4. Cano, I., Navarro-Arribas, G., Torra, V.: A new framework to automate constrained microaggregation. In: Proc. PAVLAD Workshop in CIKM 2009, Hong Kong, China, pp. 1–8. ACM, New York (2009) ISBN: 978-1-60558-884-1
5. De Waal, T.: An overview of statistical data editing. Statistics Netherlands (2008)
6. Erola, A., Castellà-Roca, J., Navarro-Arribas, G., Torra, V.: Semantic Microaggregation for the Anonymization of Query Logs. In: Domingo-Ferrer, J., Magkos, E. (eds.) PSD 2010. LNCS, vol. 6344, pp. 127–137. Springer, Heidelberg (2010)
7. Granquist, L.: The new view on editing. Int. Statistical Review 65(3), 381–387 (1997)
8. Lane, J., Heus, P., Mulcahy, T.: Data Access in a Cyber World: Making Use of Cyberinfrastructure. Transactions on Data Privacy 1(1), 2–16 (2008)
9. Lefevre, K., Dewitt, D.J., Ramakrishnan, R.: Incognito: efficient full-domain K-anonymity. In: Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data, SIGMOD 2005, pp. 49–60 (2005)
10. Martinez, S., Sánchez, D., Valls, A.: Ontology-based anonymization of categorical values. In: Torra, V., Narukawa, Y., Daumas, M. (eds.) MDAI 2010. LNCS, vol. 6408, pp. 243–254. Springer, Heidelberg (2010)
11. Martínez, S., Valls, A., Sánchez, D.: Anonymizing Categorical Data with a Recoding Method Based on Semantic Similarity. In: Hüllermeier, E., Kruse, R., Hoffmann, F. (eds.) IPMU 2010. CCIS, vol. 81, pp. 602–611. Springer, Heidelberg (2010)
12. Miller, G.: WordNet - about us. WordNet. Princeton University (2009), http://wordnet.princeton.edu
13. Navarro-Arribas, G., Torra, V.: Tree-based Microaggregation for the Anonymization of Search Logs. In: Proc. 2009 IEEE / WIC / ACM International Conference on Web Intelligence (WI 2009), Milano, Italy, September 15-18, pp. 155–158 (2009) ISBN: 978-0-7695-3801-3
14. Nin, J., Herranz, J., Torra, V.: Rethinking Rank Swapping to Decrease Disclosure Risk. Data and Knowledge Engineering 64(1), 346–364 (2008)
15. Nin, J., Herranz, J., Torra, V.: On the Disclosure Risk of Multivariate Microaggregation. Data and Knowledge Engineering 67, 399–412 (2008)
16. ODP. Open directory project (2010)
17. Pierzchala, M.: A review of the state of the art in automated data editing and imputation. In: Statistical Data Editing, Vol. 1, Conference of European Statisticians Statistical Standards and Studies N. 44, United Nations Statistical Commission and Economic Commission for Europe, 10-40 (1994)
18. Samarati, P.: Protecting Respondents' Identities in Microdata Release. IEEE Trans. on Knowledge and Data Engineering 13(6), 1010–1027 (2001)
19. Samarati, P., Sweeney, L.: Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. SRI Intl. Tech. Rep. (1998)
20. Shlomo, N., De Waal, T.: Protection of Micro-data Subjecto to Edit Constraints Against Statistical Disclousure. Journal of Official statistics 24(2), 229–253 (2008)
21. Sun, X., Li, M., Wang, H., Plank, A.: An efficient hash-based algorithm for minimal k-anonymity. In: 31st Australasian Computer Science Conference (ACSC 2008), Wollongong, NSW, Australia, January 22-25 (2008)

22. Sweeney, L.: k-anonymity: a model for protecting privacy. Int. J. of Unc., Fuzz. and Knowledge Based Systems 10(5), 557–570 (2002)
23. Torra, V.: Towards the re-identification of individuals in data files with non-common variables. In: Proc. of the 14th European Conference on Artificial Intelligence (ECAI 2000), Berlin, Germany, pp. 326–330. IOS Press, Amsterdam (2000) ISBN 1 58603 013 2
24. Torra, V.: Constrained Microaggregation: Adding Constraints for Data Editing. Transactions on Data Privacy 1(2), 86–104 (2008)
25. Torra, V.: Privacy in Data Mining. In: Data Mining and Knowledge Discovery Handbook, 2nd edn., pp. 687–716. Springer, Heidelberg (2010)
26. Torra, V.: On the Definition of Linear Constrained Fuzzy c-Means. In: Proc. of the EUROFUSE 2009, Pamplona, Spain, pp. 61–66 (September 2009) ISBN: 978-84-9769-242-7
27. Torra, V., Abowd, J.M., Domingo-Ferrer, J.: Using Mahalanobis Distance-Based Record Linkage for Disclosure Risk Assessment. In: Domingo-Ferrer, J., Franconi, L. (eds.) PSD 2006. LNCS, vol. 4302, pp. 233–242. Springer, Heidelberg (2006)
28. Torra, V., Domingo-Ferrer, J.: Record linkage methods for multidatabase data mining. In: Torra, V. (ed.) Information Fusion in Data Mining, pp. 101–132. Springer, Heidelberg (2003)
29. Torra, V., Nin, J.: Record linkage for database integration using fuzzy integrals. Int. J. of Intel. Systems 23, 715–734 (2008)
30. Verykios, V.S., Bertino, E., Nai Fovino, I., Parasiliti, L., Saygin, Y., Theodoridis, Y.: State-of-the-art in Privacy Preserving Data Mining. SIGMOD Record 33(1), 50–57 (2004)
31. Willenborg, L., de Waal, T.: Elements of Statistical Disclosure Control. Lecture Notes in Statistics. Springer, Heidelberg (2001)
32. Winkler, W.E.: Re-identification methods for masked microdata. In: Domingo-Ferrer, J., Torra, V. (eds.) PSD 2004. LNCS, vol. 3050, pp. 216–230. Springer, Heidelberg (2004)

Privacy in Data Publishing

Sabrina De Capitani di Vimercati, Sara Foresti, and Giovanni Livraga

Università degli Studi di Milano, 26013 Crema, Italia
firstname.lastname@unimi.it

Abstract. In modern digital society, personal information about individuals can be easily collected, shared, and disseminated. These data collections often contain sensitive information, which should not be released in association with respondents' identities. Removing explicit identifiers before data release does not offer any guarantee of anonymity, since de-identified datasets usually contain information that can be exploited for linking the released data with publicly available collections that include respondents' identities. To overcome these problems, new proposals have been developed to guarantee privacy in data release. In this chapter, we analyze the risk of disclosure caused by public or semi-public microdata release and we illustrate the main approaches focusing on protection against unintended disclosure. We conclude with a discussion on some open issues that need further investigation.

1 Introduction

Information is probably today one of the most valuable and demanded resources. Our personal information is collected every time we use and interact with online services (e.g., [3,8,9]). Furthermore, thanks to the advances in the Information and Communication Technologies, the storage, management, and elaboration of these large data collections is becoming easier and easier. The benefits that can be obtained sharing the collected information has attracted the interest of both public and private organizations in the data exchange and publication practices. This data sharing scenario introduces however several privacy issues. As a matter of fact, collected data often contain sensitive information that are not intended for disclosure and should therefore be adequately protected.

Many protection techniques have been proposed in the literature, depending on how collected data are released [5]. In the past, data were mainly released as *macrodata*. Macrodata consist of data that have been aggregated (e.g., the population of a county is an aggregate of the populations of the cities). Today many situations require instead the publication of *microdata*, that is, of the specific collected data related to individual *respondents*. The release of microdata instead of macrodata presents several advantages for data recipients. In fact, the availability of microdata provides higher flexibility, since it enables data recipients to perform any analysis considered of interest. However, a potential attacker can exploit linking attacks to infer sensitive information from data that are considered "sanitized" and are disclosed by other sources. Even if only in anonymous form, released microdata are still vulnerable to privacy breaches.

J. Garcia-Alfaro et al. (Eds.): DPM 2010 and SETOP 2010, LNCS 6514, pp. 8–21, 2011.

In this chapter, we illustrate the main privacy protection techniques, distinguishing between solutions designed to counteract identity disclosure and attribute disclosure. The remainder of this chapter is organized as follows. Section 2 discusses the different kinds of disclosure risk that arise in microdata release. Sections 3 and 4 illustrate the main protection techniques introduced in the literature that can counteract identity disclosure and attribute disclosure, respectively. Section 5 presents some open issues for the considered scenario. Finally, Sect. 6 concludes the chapter.

2 Protecting Privacy in Microdata Release

Microdata can be represented as tables composed of tuples defined over a set of attributes. Depending on their identifying ability and sensitivity, attributes in a microdata table can be classified in the following four categories.

- *Identifiers*: attributes that uniquely identify a respondent (e.g., `Name` and `SSN`).
- *Quasi-identifiers* (QI): attributes that, in combination, can be linked with external information to re-identify (all or some of) the respondents to whom information refers, or to reduce the uncertainty over their identities (e.g., `DoB`, `Sex`, and `ZIP`).
- *Confidential attributes*: attributes that represent sensitive information (e.g., `Disease`).
- *Non-confidential attributes*: attributes that are not considered sensitive by respondents, and whose release is not harmful (e.g., `FavoriteFood`).

Each tuple in the table is related to a unique respondent and, vice versa, each respondent is associated with one tuple in the microdata table.

The first step to protect the privacy of the respondents whose information appears in a microdata table consists in removing or encrypting explicit identifiers. This de-identification process is however not sufficient to protect respondents' identities, since the combinations of values of quasi-identifier attributes may pertain (uniquely, or almost uniquely) to specific individuals. As an example, from a study on the 2000 census data [10], Golle showed that 63% of the US population was *uniquely identifiable* by the combination of their gender, ZIP code, and full date of birth. By linking quasi-identifying attributes appearing in the de-identified microdata table with publicly available datasets, associating quasi-identifiers with respondents' identities, a malicious data recipient can possibly re-identify (or restrict the uncertainty to a specific subset of) microdata respondents and infer information that is not intended for disclosure [17].

Disclosure can be categorized as *identity disclosure* and *attribute disclosure*. Identity disclosure occurs when the identity of an individual can be reconstructed and associated with a tuple in the released microdata table. Attribute disclosure occurs when an attribute value can be associated with an individual (without necessarily being able to link the value to a specific tuple). Since datasets associating individuals' identities with their related quasi-identifying attributes are

SSN	Name	DoB	Sex	ZIP	Disease
		84/09/12	M	94139	Stomach cancer
		83/09/07	M	94139	Gastritis
		84/09/02	M	94141	Flu
		85/05/01	F	94130	Flu
		85/05/07	F	94138	Ulcera
		85/05/09	F	94138	Broken leg
		64/09/15	*M*	*94142*	Stomach cancer
		84/06/12	M	94135	Broken leg
		84/06/06	M	94137	Hypertension
		84/06/19	M	94151	Flu

(a) De-identified medical data

Name	Address	City	ZIP	DoB	Sex	Education
...	...	...	...	...	...	...
John Doe	250 Market St.	San Francisco	*94142*	*64/09/15*	*male*	secondary
...	...	...	...	...	...	...

(b) Municipality register

Fig. 1. An example of a de-identified microdata table (a) and of a publicly available non de-identified dataset (b)

often publicly available (e.g., voter registers), it is not difficult today for a data recipient to exploit this information to re-identify individuals in de-identified microdata tables. For instance, consider the de-identified microdata table in Fig. 1(a), where explicit identifiers (i.e., attributes `SSN` and `Name`) have been removed. The municipality register in Fig. 1(b) includes attributes `DoB`, `Sex`, and `ZIP`, in association with the `Name`, `Address`, `City`, and `Education` of respondents. Therefore, common attributes `DoB`, `Sex`, and `ZIP` can be exploited by a data recipient to re-identify respondents. In the microdata table in Fig. 1(a), for example, there is only one *male* born on *64/09/15* and living in *94142* area. This combination, if unique in the external world as well, uniquely identifies the corresponding tuple in the released microdata as pertaining to "*John Doe, 250 Market Street, San Francisco*", thus revealing that he suffers from *stomach cancer*.

Different factors can contribute to identity and attribute disclosure [6]. These factors include the existence of highly visible tuples (i.e., tuples with unique characteristics, such as a rare disease) and the possibility of linking the microdata table with external information sources. By contrast, there are different factors that limit the possibility to link information, such as: the natural noise characterizing the microdata table and the external sources, the presence in the table of data that are not up-to-date or that refer to different temporal intervals, and the use of different formats for representing the information in the microdata table and in the external sources. Several *microdata protection techniques* have recently been proposed to counteract improper disclosure of information [5]. These solutions aim at minimizing disclosure risks by: *i)* reducing the amount of released data; *ii)* masking the data by not releasing or perturbing their values;

iii) releasing plausible but synthetic data instead of real ones. In the remainder of this chapter, we will describe how microdata protection techniques can be adopted to prevent both identity and attribute disclosure.

3 Protecting Microdata against Identity Disclosure

We present the most popular solutions to protect released data against identity disclosure.

3.1 *k*-Anonymity

Microdata protection techniques reduce the risk of disclosure at the price of limiting the utility of the released microdata table, since they modify the data before publishing. k-anonymity [17] provides a means to characterize the degree of data protection guaranteed by the release of a de-identified microdata table. k-anonymity captures the well-known requirement, applied by statistical agencies, stating that the released data should be indistinguishably related to a certain number of different respondents. Since re-identification exploits quasi-identifier attributes, the general requirement stated above has been formulated in [17] as follows: *each release of data must be such that every combination of values of quasi-identifiers can be indistinctly matched to at least k respondents.* A microdata table satisfies the k-anonymity requirement if each tuple in the table cannot be related to less than k respondents in the population and vice versa. The satisfaction of this requirement clearly limits the possibility for a malicious data recipient to exploit quasi-identifiers to re-identify respondents. The verification of the k-anonymity requirement however requires that the data holder knows any possible external data source that contains the quasi-identifier of the microdata table. Since it seems impossible, or highly impractical, to assume such knowledge, k-anonymity takes a safe approach by requiring that, in a released microdata table, each respondent is indistinguishable from at least other $k-1$ respondents in the table itself. In practice, a microdata table is k-anonymous if each quasi-identifier value in the microdata table appears with at least k occurrences in the table itself. We note that this definition represents a sufficient condition for the k-anonymity requirement. In fact, since each combination of values of quasi-identifier attributes appears with at least k occurrences, it is not possible for a data recipient to associate a respondent with less than k tuples in the released table (and, vice versa, a tuple in the released microdata table cannot be related with less than k respondents in the population).

Many situations require that the data within each released tuple be truthful, and this makes those approaches that perturb the data before release to protect respondents' identities inapplicable. Among the microdata protection techniques proposed in the literature [5], k-anonymity relies, since its first proposal, on *generalization* and *suppression* since they guarantee the release of truthful data. *Generalization* consists in replacing the original values of attributes with more general values (e.g., the year of birth is released instead of the complete date of

Generalization	Suppression			
	Tuple	*Attribute*	*Cell*	*None*
Attribute	AG_TS	AG_AS ≡ AG_	AG_CS	AG_ ≡ AG_AS
Cell	CG_TS not applicable	CG_AS not applicable	CG_CS ≡ CG_	CG_ ≡ CG_CS
None	_TS	_AS	_CS	_ not interesting

Fig. 2. Classification of k-anonymity techniques [4]

birth). It can be applied at the level of single cell (substituting the cell value with a generalized version of it) or at the level of attribute (generalizing all the cells in the corresponding column). The cell generalization approach has the advantage of avoiding the generalization of all values in a column when generalizing only a subset of them suffices to guarantee k-anonymity. It has, however, the disadvantage of not preserving the homogeneity of the values appearing in the same column. *Suppression* consists in protecting sensitive information by removing it. Suppression, which can be applied at the level of single cell, entire tuple, or entire column, allows reducing the amount of generalization to be enforced to achieve k-anonymity. Intuitively, if a limited number of outliers would force a large amount of generalization to satisfy a k-anonymity constraint, then such outliers can be removed from the table, allowing satisfaction of k-anonymity with less generalization.

Figure 2 summarizes the different combinations of generalization and suppression at different granularity levels (including combinations where one of the two techniques is not adopted), which correspond to different approaches and solutions to the k-anonymity problem [4]. It is interesting to note that the application of generalization and suppression at the same granularity level is equivalent to the application of generalization only (AG_≡AG_AS and CG_≡CG_CS), since suppression can be modeled as a generalization to a value that represents any value in the attribute domain. Combinations CG_TS (cell generalization, tuple suppression) and CG_AS (cell generalization, attribute suppression) are not applicable since the application of generalization at the cell level implies the application of suppression at that level too. As an example, Fig. 3 illustrates a sanitized version of the table in Fig. 1(a) obtained generalizing attributes `DoB` and `ZIP` and suppressing the seventh tuple of the original microdata table. The table in Fig. 3 is k-anonymous for any $k \leq 3$, since each combination of values of quasi-identifier attributes (i.e., attributes `DoB`, `Sex` and `ZIP`) appears in the table with at least 3 occurrences. In the table, attribute `DoB` has been generalized by removing the day of birth from the complete date of birth, while attribute `ZIP` has been generalized by removing the last two digits, replaced by symbol $*$.

The application of generalization and suppression techniques produces less precise (more general) and less complete tables, while releasing truthful information that guarantees the protection of respondents' identities. The application

DoB	Sex	ZIP	Disease
1984/09	M	941**	Stomach cancer
1984/09	M	941**	Gastritis
1984/09	M	941**	Flu
1985/05	F	941**	Flu
1985/05	F	941**	Ulcera
1985/05	F	941**	Broken leg
1984/06	M	941**	Broken leg
1984/06	M	941**	Hypertension
1984/06	M	941**	Flu

Fig. 3. An example of a 3-anonymous and 3-diverse table

of generalization and suppression however causes *information loss*. It is therefore important to minimize the amount of generalization and suppression adopted to provide protection of respondents' identities. The problem of computing a k-anonymous table that minimizes the use of generalization and suppression is notably complex, regardless of the granularity at which protection techniques are applied.

Many exact and heuristic solutions have been presented in literature to compute a k-anonymous table, by applying generalization and suppression at different granularity levels. k-anonymity algorithms can also be classified on the basis of the generalization method they adopt, which can be either hierarchically-based or recoding-based. In the first case, the generalization process is driven by a predefined hierarchy of generalized values, represented as a tree, where the root corresponds to the most general attribute value, and the leaves to the most specific values. Two well known exact algorithms adopting a hierarchy-based generalization are Samarati's algorithm [17] and the Incognito algorithm [12], both applying attribute level generalization and tuple level suppression. Recoding-based generalization partitions the domain of the attribute to be generalized in a set of disjoint intervals, and each original value in the microdata table is generalized substituting it with the interval it belongs to. Examples of algorithms adopting recoding-based generalization techniques are k-Optimize [1], which is an exact algorithm in AG_TS class, Mondrian multidimensional k-anonymity [13], which is an exact algorithm in CG_ class, and the heuristic algorithm presented in [11], which belongs to AG_TS class.

3.2 k-Anonymity Variations

Recently, new techniques have been proposed to protect respondents' identities from disclosure. These solutions are based on the definition of more specific and fine-grained metrics than k-anonymity that are specifically suited to particular scenarios with peculiar privacy threats.

(X, Y)-anonymity. The assumption underlying microdata release that each respondent is represented in the microdata table by a single tuple may not fit many real world scenarios. In these situations, k-anonymity may not guarantee

the privacy of the respondents. In fact, a set of tuples with the same value for the quasi-identifier attributes could all be related to less than k respondents, thus violating the k-anonymity requirement. To overcome this problem, Wang and Fung propose (X,Y)-anonymity [18]. The k-anonymity requirement is enforced in this scenario by the definition of (X,Y)-anonymity formulated as follows. Given the set Y of identifiers and the set X of quasi-identifiers in a microdata table, the table is (X,Y)-anonymous if there are at least k distinct values for Y attributes for each combination of quasi-identifying X attributes. Intuitively, a table is (X,Y)-anonymous if each group of tuples sharing the same quasi-identifier value is associated with no less than k different respondents. We note that explicit identifiers in Y are removed from the table before publication.

Multi-relational k-anonymity. While traditional microdata protection techniques assume the release of a unique microdata table, this assumption results limiting in different scenarios. Multi-relational k-anonymity addresses the problem of guaranteeing the protection of the respondents' identities when multiple tables are released [16]. The set of released tables consists of a *person-specific* table PT containing personal identifiable information and sensitive attributes, and a set $T_1, \ldots, T_n$ of tables containing sensitive attributes, quasi-identifying attributes, and foreign keys that refer to attributes in PT. Multi-relational k-anonymity basically translates the (X,Y)-anonymity condition to operate on the join $PT \bowtie T_1 \bowtie \ldots \bowtie T_n$ of the released tables. A microdata table fulfills multi-relational k-anonymity if, in the result of the join, each group of tuples sharing the same quasi-identifier value refers to no less than k different respondents.

4 Protecting Microdata against Attribute Disclosure

Although k-anonymity and its variations illustrated in the previous section guarantee to protect respondents' identities, from a k-anonymous table it is still possible to identify, or reduce the uncertainty about, the sensitive information related to a specific respondent. In other words, protection against *identity disclosure* does not imply protection against *attribute disclosure*. In the following, we present the most popular solutions to protect released data against attribute disclosure.

4.1 ℓ-Diversity

Machanavajjhala, Gehrke, and Kifer define two possible attacks to k-anonymous tables that put sensitive attributes at risk: *homogeneity attack* (already noted in [17]) and *external knowledge attack* [15]. Consider a k-anonymous table, where there is a sensitive attribute and suppose that all the tuples with a specific value for the quasi-identifier (referred to as *equivalence class*) have the same value for the sensitive attribute as well. Under this homogeneity assumption, if an attacker knows the quasi-identifier value of a respondent and knows that this respondent belongs to the population represented in the table, the attacker can infer which is the value for the sensitive attribute for the known respondent. As

an example, consider an equivalence class where all the tuples have *cancer* as `Disease`, and suppose that *Alice* knows that *Bob* is in this equivalence class. Alice can then infer that *Bob* suffers from *cancer*. External knowledge attack is based on a priori knowledge of an attacker of some additional information, allowing her to reduce her uncertainty about respondents' sensitive attribute values. For instance, consider an equivalence class where two out of three tuples have *cancer* as `Disease`, and the third tuple has *broken leg*. Suppose that *Alice* knows that *Bob* is in that equivalence class and that she has seen him running (external knowledge). *Alice* can then infer that *Bob* cannot have a broken leg, and therefore he suffers from *cancer*.

To counteract attribute disclosure caused by homogeneity and external knowledge attacks, Machanavajjhala et al. [15] introduce the notion of ℓ-*diversity*. ℓ-diversity requires that, in each equivalence class, there exist at least ℓ *well-represented* values for the sensitive attribute. In [15], the authors provide several definitions for "well-represented" values. A straightforward definition states that ℓ values are well-represented if they are different. Therefore, a microdata table is ℓ-diverse if each equivalence class has at least ℓ different values for the sensitive attribute. It is easy to see that, in a ℓ-diverse table, homogeneity attack is no more applicable, since each equivalence class has at least ℓ different values for the sensitive attribute. Also, external knowledge attack becomes more complicated as ℓ increases, since the attacker needs more knowledge to individuate a unique sensitive attribute value that can be associated with a particular respondent. As an example, the 3-anonymous table in Fig. 3 is also 3-diverse. Suppose that *Alice* knows only that *Bob* belongs to the last equivalence class, and that he does not have a broken leg. *Alice* can now infer that *Bob* suffers either from *flu* or from *hypertension*, with equal probability. The problem of determining an optimal ℓ-diverse table is computationally hard [15], and the authors suggest to adopt k-anonymity algorithms to deal with the ℓ-diversity requirement.

4.2 *t*-Closeness

While representing a first step to counteract attribute disclosure, ℓ-diversity may not be sufficient to provide proper protection to the sensitive information in a microdata table. Machanavajjhala et al. define two possible attacks to ℓ-diverse tables: *skewness attack* and *similarity attack* [14]. Skewness attack occurs when the distribution of values of the sensitive attribute within a given equivalence class is different from the distribution of the values for the same attribute over the whole population (or the microdata table). As an example, consider an equivalence class containing three tuples in a 2-diverse table, and suppose that two of these tuples have *H1N1* and the other tuple has *flu* as values for attribute `Disease`. Since *H1N1* is a rare disease that occurs with a probability much lower than *flu*, the equivalence class reveals that its respondents have a higher probability of suffering from *H1N1*. If *Alice* knows that *Bob* belongs to that equivalence class, she can infer that *Bob* has 67% probability of suffering from *H1N1*. Similarity attack occurs when the values of the sensitive attribute for an equivalence class are, although different, semantically similar. In this case,

DoB	Sex	ZIP	Disease
1984	M	941**	Broken leg
1984	M	941**	Gastritis
1984	M	941**	Flu
1985	F	941**	Flu
1985	F	941**	Ulcera
1985	F	941**	Broken leg
1984	M	941**	Stomach cancer
1984	M	941**	Hypertension
1984	M	941**	Flu

Fig. 4. An example of a 3-anonymous and 3-diverse table that respects t-closeness

an attacker can learn important information about the value of the sensitive attribute associated with a respondent in the equivalence class. For instance, consider the table in Fig. 3 and suppose that *Alice* knows that *Bob* is a male, born in September 1984. *Bob* is then represented by a tuple in the first equivalence class. Even if this equivalence class is 3-diverse, two tuples out of three are associated with stomach related diseases. Therefore, *Alice* can infer that *Bob* has 67% probability of suffering from stomach related diseases.

The privacy issues described above arise because ℓ-diversity considers neither the semantics of values in equivalence classes, nor the information that can be inferred from statistical analysis over the released table. To address these problems, Machanavajjhala et al. [14] propose the notion of t-closeness. t-closeness extends the definition of k-anonymity by requiring that the value distribution of the sensitive attribute in each equivalence class be close (with a difference lower than threshold t) to the value distribution of the same attribute in the released microdata table. In this way, skewness attack is no more applicable, since t-closeness reduces the difference between the value distribution of the sensitive attribute in the population and the value distribution of the sensitive attribute in each equivalence class. Also, similarity attack becomes more complicated, since the presence of semantically similar values in an equivalence class does not give additional information to recipients with respect to the knowledge of the whole microdata table.

To illustrate, consider the table in Fig. 4, where attribute `DoB` has been generalized by releasing only the year of birth, attribute `ZIP` has been generalized by removing the last two digits, and the seventh tuple has been suppressed. In each equivalence class the values of the sensitive attribute `Disease` are different and not semantically similar: it is easy to see that both skewness attack and similarity attack are no more applicable.

The enforcement of t-closeness implies the ability of computing the difference between the value distribution of the sensitive attribute over the whole table and the value distribution of the sensitive attribute in each equivalence class. In [14], the authors present different techniques to measure the difference between value distributions, such as the *Earth Mover Distance* (EMD).

4.3 Further Proposals for Protecting Microdata against Attribute Disclosure

Besides ℓ-diversity and t-closeness, recently alternative solutions have been proposed to protect microdata releases against attribute disclosure.

(α_i, β_i)-closeness. The definition of t-closeness, as highlighted in [7], is based on the limiting assumption that all the values assumed by the sensitive attribute are equally sensitive. Also, t-closeness is a difficult property to achieve, since choosing an appropriate value for the threshold t is not intuitive and a low value for threshold t implies the release of a useless table. To address these issues, in [7] the authors extend the definition of t-closeness by including a range $[\alpha_i, \beta_i]$ for each value in the sensitive attribute domain. An equivalence class satisfies (α_i, β_i)-closeness if the percentage of tuples in the equivalence class having the same sensitive value v_i is higher than α_i and lower than β_i, where $[\alpha_i, \beta_i]$ is the range associated with value v_i. A microdata table satisfies (α_i, β_i)-closeness if each equivalence class in the table satisfies (α_i, β_i)-closeness. The definition of a range $[\alpha_i, \beta_i]$, instead of a unique threshold t, allows the data publisher to specify different sensitivity levels for different sensitive values. Also, the definition of a range is more intuitive than the definition of a unique threshold, since values α_i and β_i represent the lower and upper bounds to the risk of inferring the specific sensitive value, and the quality of the released data increases. In fact, only sensitive values are subject to restrictive thresholds, while non-sensitive values can be released adopting less limiting protection measures.

Confidence bounding. Traditionally, protection techniques designed to counteract attribute disclosure assume that any association between the quasi-identifier and the sensitive attribute needs to be protected. Also, these solutions assume that any combination of values of quasi-identifying attributes causes the same risk of disclosure and that all the values in the domain of the sensitive attribute are equally sensitive. However, in many real world scenarios, these assumption may not hold. A more precise description of the disclosure risk (and therefore of the privacy protection requirement) has been recently proposed in [19,20], by the definition of *privacy templates*. A privacy template is a rule of the form $\langle QI \rightarrow s, h \rangle$, where QI is a quasi-identifier, s is a value in the domain of the sensitive attribute, and h is a privacy threshold. It states that a microdata table can be safely released if any data recipient cannot infer with probability higher than $h\%$ that the value assumed by the sensitive attribute for a given tuple is s, given the values assumed by the attributes in QI. Intuitively, if at least one of the association rules between the values of the quasi-identifier and s has confidence higher than h in the released microdata table, the privacy template is violated. For instance, privacy template $\langle \{\text{DoB,ZIP}\} \rightarrow H1N1, 10\% \rangle$ is violated if the confidence of association rule $\{1975,22010\} \rightarrow H1N1$ is higher than 10% (i.e., more than the 10% of the tuples in the released table have `DoB`=1975, `ZIP`=22010 and `Disease`=H1N1). A microdata table can therefore be released without risk of attribute disclosure only if it satisfies all the privacy templates defined for it.

(X,Y)-privacy. As already noted, one of the main limitations of microdata protection techniques is that they assume that each respondent is represented by a unique tuple in the microdata table. To overcome this limitation, Wang and Fung [18] propose to reformulate the definition of k-anonymity through the more general concept of (X,Y)-anonymity (Sect. 3). Also, the authors introduce (X,Y)-privacy as a means to provide protection against attribute disclosure when more tuples in the microdata table are possibly related to the same respondent. Given the set Y of identifiers and the set X of quasi-identifiers in a microdata table, the table satisfies (X,Y)-privacy if it is (X,Y)-anonymous and it is not possible for a data recipient to infer with probability higher than a fixed threshold h the value of the sensitive attribute associated with a given respondent.

Personalized privacy. A common assumption on which traditional microdata protection techniques rely on is that all the respondents represented in a microdata table enjoy the same privacy degree, chosen before publication. However, in some cases respondents may have different requirements on the privacy degree they want for the tuple representing their data in the released table. Xiao and Tao [21] put forward the idea that each respondent can specify her privacy preferences. Personalized privacy assumes that each sensitive attribute has a *taxonomy tree* (e.g., a generalization hierarchy), and that each respondent specifies a *guarding node* in this tree. The privacy of a data respondent is exposed if it is possible for a data recipient to infer a sensitive value belonging to the subtree of her guarding node (i.e., a more specific value) with a probability greater than a chosen threshold. For instance, suppose that $H1N1$ and Flu are child nodes of value *ViralDisease* in the taxonomy tree of sensitive attribute `Disease`. Suppose that a patient suffering from $H1N1$ wants her disease to be protected, whereas a patient suffering from Flu does not. In this case, the first patient may choose *ViralDisease* as guarding node, and the second patient may not choose any guarding node since she does not care about revealing her illness. Personalized privacy allows data publisher to take into consideration respondents' privacy requirements, providing greater flexibility in the anonymization process.

5 Open Issues

Despite the wide attention devoted to the problem of preventing identity and attribute disclosure in microdata release, there are still several issues that need to be further investigated. We discuss some of them in the following.

Privacy metrics. Different techniques have been proposed in the literature to protect microdata releases against identity and/or attribute disclosure. These techniques offer different protection guarantees that, unfortunately, cannot be directly compared. As an example, it is not possible to compare a 5-anonymous table with a 3-diverse table, since the two parameters considered (i.e., k and ℓ) do not represent the same privacy measure. To permit the data owner to identify the protection technique that better suits her needs, it is therefore necessary to define a unique privacy metric.

Data utility measures. The adoption of microdata protection techniques needs to balance two contrasting needs: the need of the data recipient for complete and detailed data, and the need of respondents for privacy protection. Recently, several techniques have been proposed to assess the information loss (and, consequently, the utility) due to the application of a given protection technique. However, current proposals are dependent on the actual protection model that is applied to the released dataset. Moreover, it has to be noted that the utility of a given dataset is not an absolute measure: it is closely related to how data will be used. For instance, an anonymized dataset revealing only patients' date of birth and illnesses could be useful for the pharmaceutical industry, while it could be useless for the government tax agency, who is interested in the association of illnesses with the cost of treatments. The definition of utility metrics should therefore take into account the purpose of data release.

External knowledge. In today's society, a data recipient can obtain information about the respondents represented in a published dataset. This external knowledge can come, for example, from similar collections of data released by other organizations or competitors, social networks, or personal knowledge. All these external information sources can possibly be exploited by malicious data recipients to determine (or reduce her uncertainty on) the identity of data respondents and/or the sensitive attributes associated with them. Recently, novel data protection techniques are taking external knowledge into consideration in microdata publication [2]. However, these solutions are limited since they only consider a specific kind of external knowledge (i.e., the personal knowledge of a respondent and the distribution of sensitive values in the population). The definition of a comprehensive modeling of the external knowledge is however necessary for the definition of protection techniques able to effectively counteract external knowledge attacks.

Data dependencies. Microdata protection techniques are based on the assumption that the attributes in the released table are not correlated. However, this assumption does not fit many real world scenarios. Also, the presence of dependencies and correlations among published data can be exploited for inferring sensitive information which is not intended for release. As an example, suppose that attribute `Disease` has been removed from a microdata table before publishing and that attribute `Treatment` has been released. Since there is a dependency between the two attributes, a recipient may easily infer the disease of a respondent by knowing the treatment with which she is being cared. The definition of privacy techniques should therefore be able to model dependencies among data, to the aim of preventing the unintentional disclosure of sensitive information that are not explicitly released, but that can be easily inferred from the data.

Longitudinal data. Traditional microdata protection techniques assume the released dataset to be static. However, in different scenarios the data owner may need to release different observations over time of the same data, that is, *longitudinal data*. For instance, a hospital may need to publish, at regular time intervals, data collections that represent the evolution of patients' illnesses. Longitudinal data release poses new issues on the problem of privacy preservation,

since anonymizing each dataset without considering previous releases of the same data does not provide guarantees of proper data protection. Indeed, information referred to the same respondent appears in different versions of the data and could be possibly exploited to identify her. Privacy protection of longitudinal data should guarantee that the combined knowledge of different versions of the same dataset does not cause identity and attribute disclosure.

Multiple views. Current proposals for data protection techniques assume the existence of a unique static table that is fully released at one time. In different scenarios, a data holder may be interested in releasing different views (subsets) of her original data collection. The publication of different views of the dataset, if not performed carefully, may cause privacy breaches, since these views can possibly be joined, thus revealing information that was not intended for disclosure. The design of privacy protection techniques for the release of multiple views over a unique dataset should then take into consideration any possible join (either exact or loose) that data recipients can evaluate over the released views, to prevent both identity and attribute disclosure.

6 Conclusions

Data dissemination and sharing are becoming increasingly popular practices. Although in the past data were mostly released in tabular form, many situations require today the release of specific microdata, thus putting respondents' privacy at risk. To address this issue, a variety of data protection techniques have been proposed in the literature. In this chapter, we first discussed the different kinds of disclosure risks that can arise in microdata publication. We then illustrated the main privacy protection techniques, distinguishing between solutions aimed at protecting microdata against identity and attribute disclosure. We finally introduced some open issues that deserve further investigations.

Acknowledgments

This work was supported in part by the EU (project "PrimeLife", 216483); Italian MIUR (project "PEPPER" 2008SY2PH4); UNIMI (internal project "Privacy-aware environmental data publishing").

References

1. Bayardo, R., Agrawal, R.: Data privacy through optimal k-anonymization. In: Proc. of the 21st International Conference on Data Engineering (ICDE 2005), Tokyo, Japan (2005)
2. Chen, B., LeFevre, K., Ramakrishnan, R.: Privacy skyline: Privacy with multidimensional adversarial knowledge. In: Proc. of the 33rd International Conference on Very Large Data Bases (VLDB 2007), Vienna, Austria (2007)
3. Cimato, A., Gamassi, M., Piuri, V., Sassi, R., Scotti, F.: Privacy-aware biometrics: Design and implementation of a multimodal verification system. In: Proc. of Annual Computer Security Applications Conference (ACSAC 2008), Anaheim, USA (2008)

4. Ciriani, V., De Capitani di Vimercati, S., Foresti, S., Samarati, P.: k-Anonymity. In: Yu, T., Jajodia, S. (eds.) Secure Data Management in Decentralized Systems. Springer, Heidelberg (2007)
5. Ciriani, V., De Capitani di Vimercati, S., Foresti, S., Samarati, P.: Microdata protection. In: Yu, T., Jajodia, S. (eds.) Secure Data Management in Decentralized Systems. Springer, Heidelberg (2007)
6. Federal Committee on Statistical Methodology: Report on statistical disclosure limitation methodology. Statistical Policy Working Paper 22, USA (1994)
7. Frikken, K., Zhang, Y.: Yet another privacy metric for publishing micro-data. In: Proc. of the 7th ACM Workshop on Privacy in Electronic Society (WPES 2008), Alexandria, VA, USA (2008)
8. Gamassi, M., Lazzaroni, M., Misino, M., Piuri, V., Sana, D., Scotti, F.: Accuracy and performance of biometric systems. In: Proc. of IEEE Instrumentation and Measurement Technology Conference (IMTC 2004), Como, Italy (2004)
9. Gamassi, M., Piuri, V., Sana, D., Scotti, F.: Robust fingerprint detection for access control. In: Proc. of RoboCare Workshop 2005, Rome, Italy (2005)
10. Golle, P.: Revisiting the uniqueness of simple demographics in the US population. In: Proc. of the 5th ACM Workshop on Privacy in Electronic Society (WPES 2006), Alexandria, VA, USA (2006)
11. Iyengar, V.: Transforming data to satisfy privacy constraints. In: Proc. of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2002), Alberta, Canada (2002)
12. LeFevre, K., DeWitt, D., Ramakrishnan, R.: Incognito: Efficient full-domain k-anonymity. In: Proc. of the 31st ACM SIGMOD International Conference on Management of Data (SIGMOD 2005), Baltimore, MA, USA (2005)
13. LeFevre, K., DeWitt, D., Ramakrishnan, R.: Mondrian multidimensional k-anonymity. In: Proc. of the 22nd International Conference on Data Engineering (ICDE 2006), Atlanta, GA, USA (2006)
14. Li, N., Li, T., Venkatasubramanian, S.: t-closeness: Privacy beyond k-anonymity and ℓ-diversity. In: Proc.of the 23rd International Conference on Data Engineering (ICDE 2007), Istanbul, Turkey (2007)
15. Machanavajjhala, A., Kifer, D., Gehrke, J., Venkitasubramaniam, M.: ℓ-diversity: Privacy beyond k-anonymity. ACM Transactions on Knowledge Discovery from Data 1(1), 3:1–3:52 (2007)
16. Nergiz, M., Clifton, C., Nergiz, A.: Multirelational k-anonymity. In: Proc. of the 23rd International Conference on Data Engineering (ICDE 2007), Istanbul, Turkey (2007)
17. Samarati, P.: Protecting respondents' identities in microdata release. IEEE Transactions on Knowledge and Data Engineering 13(6), 1010–1027 (2001)
18. Wang, K., Fung, B.: Anonymizing sequential releases. In: Proc. of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2006), Philadelphia, PA, USA (2006)
19. Wang, K., Fung, B.C.M., Dong, G.: Integrating private databases for data analysis. In: Kantor, P., Muresan, G., Roberts, F., Zeng, D.D., Wang, F.-Y., Chen, H., Merkle, R.C. (eds.) ISI 2005. LNCS, vol. 3495, pp. 171–182. Springer, Heidelberg (2005)
20. Wang, K., Fung, B., Yu, P.: Handicapping attacker's confidence: An alternative to k-anonymization. Knowledge and Information Systems 11(3), 345–368 (2007)
21. Xiao, X., Tao, Y.: Personalized privacy preservation. In: Proc. of the 32nd ACM SIGMOD International Conference on Management of Data (SIGMOD 2006), Chicago, IL, USA (2006)

A User-Oriented Anonymization Mechanism for Public Data

Shinsaku Kiyomoto and Toshiaki Tanaka

KDDI R & D Laboratories Inc.
2-1-15 Ohara, Fujimino-shi, Saitama, 356-8502, Japan
{kiyomoto,toshi}@kddilabs.jp

Abstract. A challenging task in privacy protection for public data is to realize an algorithm that generalizes a table according to a user's requirement. In this paper, we propose an anonymization scheme for generating a k-anonymous table, and show evaluation results using three different tables. Our scheme is based on full-domain generalization and the requirements are automatically incorporated into the generated table. The scheme calculates the scores of intermediate tables based on user-defined priorities for attributes and selects a table suitable for the user's requirements. Thus, the generated table meets user's requirements and is employed in the services provided by users without any modification or evaluation.

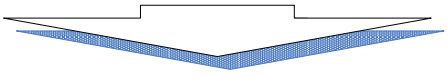
1 Introduction

Privacy is an increasingly important aspect of data publishing. Sensitive data, such as medical records in public databases, are recognized as a valuable source of information for the allocation of public funds, medical research and statistical trend analysis [1]. However, if personal private information is leaked from the database, the service will be regarded as unacceptable by the original owners of the data[11]. Thus, anonymization methods have been considered as a possible solution for protecting privacy information[9]. One class of models, called *global-recoding*, maps the values of attributes to other values [3,4] in order to generate an anonymized dataset. This paper focuses on a specific global-recoding model "*full-domain generalization*". Generally, some anonymization tables are generated from an original table and users select a table from the anonymization tables based on certain requirements for the services that they provide to the public. Figure 1 shows an example case; the attribute "Zip" and "Nationality" are generalized to satisfy 2-anonymity. A challenging issue in the anonymization of tables is to realize an algorithm that generalizes a table according to a user's requirement. For example in Figure 1, it is possible to generalize "Gender" and hold "Nationality" information for a 2-anonymization table, where the user hope to obtain detailed information of "Nationality". If the algorithm incorporates the requirements into a generated table and outputs the most suitable table, evaluation and selection of candidates for anonymization tables are not needed when using an anonymization table.

J. Garcia-Alfaro et al. (Eds.): DPM 2010 and SETOP 2010, LNCS 6514, pp. 22–35, 2011.

Original Table

Quasi-Identifiers				Sensitive Information
Birth	Gender	Zip	Nationality	Problem
1985	Male	01243	UK	Shortness of Breath
1985	Male	01242	Italy	Chest Pain
1985	Female	01241	UK	Hypertension
1985	Female	01245	Italy	Hypertension
1984	Male	01234	USA	Chest Pain
1984	Male	01232	USA	Obesity



Example of Anonymization Table (k=2)

Quasi-Identifiers				Sensitive Information
Birth	Gender	Zip	Nationality	Problem
1985	Male	0124*	Europe	Shortness of Breath
1985	Male	0124*	Europe	Chest Pain
1985	Female	0124*	Europe	Hypertension
1985	Female	0124*	Europe	Hypertension
1984	Male	0123*	USA	Chest Pain
1984	Male	0123*	USA	Obesity

Fig. 1. Example of Anonymization Table

In this paper, we propose a k-anonymization mechanism that reflects the user's requirements. The mechanism evaluates the score of a table for each iteration of generalization and selects the best score table for the next iteration. After some iterations, the mechanism provides a k-anonymous table that is suitable for the user's requirements. Properties of our mechanism are summarized as follows;

- The mechanism is constructed based on combination of top-down and bottom-up algorithms for full-domain generalization, and outputs an anonymization table that has the best score in the execution.
- A user inputs not only constraints of generalization boundaries but also priorities for quasi-identifiers as user's requirements on the anonymization table.
- The mechanism allows the modification of the anonymity check logic and the score function.

2 Related Work

There are two major approaches to avoiding leaks of private information from public databases: perturbative methods and non-perturbative methods. Generalization methods are non-perturbative methods and they modify the original data to avoid identification of the records. These methods generate a common value for some records and replace identifying information in the records with the common value. However, detailed information is lost during this process. On the other hand, perturbative methods add noise to data. While perturbed data usually retains detailed information, it also normally includes fake data.

Differential Privacy [12,13] is a notion of privacy for perturbative methods that is based on the statistical distance between two database tables differing by at most one element. The basic idea is that, regardless of background knowledge, an adversary with access to the data set draws the same conclusions, whether or not a person's data is included in the data set. That is, a person's data has an insignificant effect on the processing of a query. Differential privacy is mainly studied in relation to perturbation methods [16,35,14,15] in an interactive setting, although it is applicable to a certain generalization method [18].

Samarati and Sweeney [29,28,31] proposed a primary definition of privacy that is applicable to generalization methods. A data set is said to have *k-anonymity* if each record is indistinguishable from at least $k-1$ other records with respect to certain identifying attributes called *quasi-identifiers* [10]. In other words, at least k records must exist in the data set for each combination of the identifying attributes. Clearly any generalization algorithm that converts a database into one with k-anonymity involves a loss of information in that database.

Minimizing this information loss thus presents a challenging problem in the design of generalization algorithms. The optimization problem is referred to as the k-anonymity problem. Meyerson reported that optimal generalization in this regard is an NP-hard problem[26]. Aggarwal *et al.* proved that finding an optimal table including more than three attributes is NP-hard [3]. Nonetheless, k-anonymity has been widely studied because of its conceptual simplicity [5,24,25,36,33,30]. Machanavajjhala *et al.* proposed another important definition of privacy in a public database [24]. The definition, called l-diversity assumes a strong adversary having certain background knowledge that allows the adversary to identify object persons in the public database.

Samarati proposed a simple binary search algorithm for finding a k-anonymous table[28]. A drawback of Samarati's algorithm is that for arbitrary definitions of minimality, it is not always guaranteed that this binary search algorithm can find the minimal k-anonymity table. Sun *et. al.* presented a hash-based algorithm that improves the search algorithm[30]. Aggarwal *et al.* proposed $O(k)$-approximation algorithm [4] that for the k-anonymity problem. A greedy approximation algorithm [20] proposed by LeFevre *et al.* searches optimal multi-dimensional anonymization. A genetic algorithm framework [17] was proposed because of its flexible formulation and its ability to find more efficient anonymization. Utility-based anonymization [39,38] makes k-anonymous tables using a heuristic local recoding anonymization. Moreover, the k-anonymization problem is viewed as a clustering problem. Clustering-based approaches [8,32,22,40] search a cluster that has k-records. In full-domain generalization, there are two heuristic approaches for generalization algorithms: the top-down approach and the bottom-up approach. Bayardo and Aggrawal proposed a generalization algorithm using the top-down approach [7]. The algorithm finds a generalization that is optimal according to a given fixed cost metric for a systematic search strategy, given generalization hierarchies for a single attribute. Incognito [19] is a bottom-up-based algorithm that produces all possible k-anonymous tables for an original table.

There are a few research papers about k-anonymization based on requirements of users that need anonymized public data. Loukides *et al.* considered a k-anonymization approach [23] according to both the data owner's policies and a user's requirement. Aggarwal and Yu discussed a condensation based approach [2] for different privacy requirements. LeFevre *et al.* provides an anonymization algorithm [21] that produces an anonymous view depending on data mining tasks. Xiao and Tao proposed a concept of *personalized anonymity* and presented an generalization method [37] that performs the minimum generalization for requirements of data owners. Miller *et al.* presented an anonymization mechanism [27] that provides k-anonymous tables under generalization boundaries of quasi-identifiers. The configuration of the boundaries can be considered to be a user's requirement. However, to reflect a user's requirement precisely, the data situation, such as the number of attribute values, needs to be considered in the anonymization of the table.

3 Preliminary

In this section, the definitions and assumptions are introduced.

3.1 *k*-Anonymity

A database table T in which the attributes of each user are denoted in one record is in a public domain and an attacker obtains the table and tries to distinguish the record of an individual. Suppose that a database table T has m records and n attributes $\{A_1, \ldots, A_n\}$. Each record $\mathbf{a^i} = (a_1^i, \ldots, a_n^i)$ can thus be considered as an n-tuple of attribute values, where a_j^i is the value of attribute A_j in record $\mathbf{a^i}$. The database table T itself can thus be regarded as the set of records $T = \{\mathbf{a^i} : 1 \leq i \leq m\}$. The definition of k-anonymity is as follows;

k-Anonymity [28]. A table T is said to have *k-anonymity* if and only if each n-tuple of attribute values $\mathbf{a} \in T$ appears at least k times in T.

3.2 Full-Domain Generalization

Full-Domain Generalization for obtaining a k-anonymous table consists of replacing attribute values with a generalized version of these values and it is based on generalization hierarchies [9]. A quasi-identifier is an attribute that can be joined with external information to re-identify individual records with sufficiently high probability [10]. Generally, a target table $T^x = (T^q|T^s)$ consists of two types of information: a subtable of quasi-identifiers T^q and a subtable of sensitive information T^s. Since the sensitive attributes represent the essential information with regard to database queries, a generalization method is used to modify (anonymize) T^q in order to prevent the identification of the owners of the sensitive attributes, while retaining the full information in T^s. Thus, the generalization algorithm focuses on the subtable of quasi-identifier T^q and modifies it to satisfy k-anonymity. We assume that quasi-identifier attributes are known for the generalization algorithm.

3.3 Generalization Hierarchies

As in previous work in full-domain generalization, we assume that a generalization hierarchy H_{a^i} is defined for each attribute a^i. The generalization algorithm understands generalized attribute values according to the generalization hierarchies and modifies each attribute using them. Each attribute value in the hierarchy has a generalization level w_{a^i}. The level of the top node is 0 and generally $w_{a^i}+1$ level nodes include more detailed information than nodes of level w_{a^i}. Furthermore, the number $b_{w_{a^i}}$ of nodes is defined for each level w_{a^i}. The symbol $b_{w_{a^i}}$ denotes the number of possible types for values of the attribute a^i.

3.4 Top-Down and Bottom-Up

There are two methods for generating k-anonymous tables in generalization schemes: the top-down approach and the bottom-up approach. The top-down approach starts at the root table where all attributes have a root value (a maximally generalized value), and finds a k-anonymous table to change attribute values to lower values (more detailed values) of the generalization hierarchy. On the other hand, the initial table of the bottom-up approach is an original table and attribute values are replaced using upper attribute values until a k-anonymous table is found. Our scheme uses a top-down approach as a basic algorithm to reduce the number of score calculations. Furthermore, we consider a pre-computation in the top-down approach in order to skip some computations by starting at the root table. The details of our scheme are described in a later section.

3.5 Service Model

We assume a service model as in Figure 2. A data holder has an original table including sensitive privacy information, and provides a service that generates

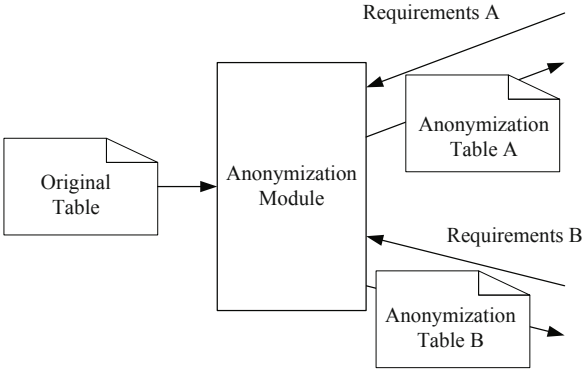


Fig. 2. Service Model

an anonymization table based on a user's requirement and supplies it to the user. The anonymization mechanism generates different tables for any users; indeed one user's requirement is generally different from that of other users. Thus, a user-oriented anonymization mechanism is required for the service. We assume that requirements from owners of records in the table is defined as k of k-anonymity.

4 *k*-Anonymization Mechanism

We present our k-anonymization mechanism in this section. First, we explain the input by a user; the information is used to reflect the user's requirement. Next, we present our basic algorithm, and an extension of the algorithm to obtain a sufficient table.

4.1 Requirements

In our system, a user can input the following conditions for k-anonymization.

- *Priority.* The user defines a positive integer value v_{a^i} for each attribute a^i. The value depends on the priority of the attribute. That is, $v_{a^i} = mv_{a^j}$, where the priority of the attribute a^i is m-times higher than a^j. For example, the user can define $(v_{a^1}, v_{a^2}, v_{a^3}, v_{a^4}) = (10, 5, 1, 1)$. The user configures high priority for an attribute where the user desires more detailed information about the attribute.
- *Minimum Level.* The user can define a minimum level for each attribute. Each attribute has a hierarchical tree structure. The minimum level $w^M_{a^i}$ defined by the user means that a k-anonymized dataset generated by the system includes at least $w^M_{a^i}$-level information for the attribute. The system does not generalize the attribute below the minimum level.

The above two requirements reflect the granularity of information in a generated k-anonymous table. The mechanism tries to keep attribute values located at the lower node as much as possible in the generalization hierarchy while satisfying k-anonymity, where the user marks the attribute as high priority. Furthermore, the user controls the limits of generalization using a configuration of minimum levels for attributes.

4.2 Functions

The mechanism generates a k-anonymous table T^G and calculates its score s based on input data: a table $T = T^q$ that consists of m records and n quasi-identifiers a^i (i=1, ...,n), parameters k, generalization hierarchy for the attributes H_{a^i}, lowest levels $w^L_{a^i}$ of H_{a^i} and user's requirements v_{a^i} and $w^M_{a^i}$. The parameters k is the requirement for privacy (which means k-anonymity) and a system parameter that is defined according to the target table. The score s denotes a rating of the degree to which the user's requirements are satisfied. The following subfunctions are used in the algorithm:

- Sort$(T, v_{a^1}, ..., v_{a^n})$. This function sorts attributes a^i $(i = 1, ..., n)$ in the Table T by user-defined priority values v_{a^i}. The function generates $T = (a^1, ..., a^n)$ in an order whereby v_{a^1} is the smallest and v_{a^n} is the largest.
- Check$_k(T)$. This function calculates a minimum number of group members, where $T = (T_q|T_s)$ and all records in T_q are categorized into groups with the same attribute values. That is, this function calculates x of x-anonymity for T_q. First, the function generates a hash value of each record in the table T_q and counts the number of hash values that are the same. If the numbers of all hash values are more than x, the function outputs x. This process is a simple and efficient way for checking k-anonymity and is similar to the process adopted in the previous work. This function is implemented as a modifiable module; we can add other checking logics of anonymity definitions such as l-diversity [24].
- Generalization(a^i, H_{a^i}, w_{a^i}). This function modifies the attribute a^i based on its generalization hierarchy H_{a^i}. The attribute values change to upper node values of level w_{a^i-1}, where the level of the attribute value is w_{a^i}.
- De-Generalization(a^i, H_{a^i}, w_{a^i}). This function modifies the attribute a^i based on its generalization hierarchy H_{a^i}. The attribute values change to lower node values of level w_{a^i+1}, where the level of the attribute value is w_{a^i}.
- Score (T). This function calculates the score of a table T. Our system calculates the score S_t of the anonymized datasets using the following equation.

$$S_t = \sum_{\forall a^i} v_{a^i} \cdot e_{a^i}$$

where e_{a^i} is a number for the type of values of an attribute a^i in the table. The score is high where a user-defined priority value for the attribute is high and the attribute has many types of values. The function produces a single value; thus, different priority values may produce the same value. The function is implemented as a replaceable module; thus, we can adjust the function or add another score function according to data types. Note that the table of the best score is not optimal in k-anonymous tables, but suitable for user's requirements.

4.3 Basic Algorithm

In the generated table, information that is important for the user is maintained at a level that is as detailed as possible, while other information remains at the level of generalized information. The algorithm consists of two main parts: pre-computation and top-down generation.

Pre-computation. The mechanism uses pre-computation to reduce the total computation cost of k-anonymization. The pre-computation consists of two steps; the step 1 consider single attribute anonymity and generalize each attribute to satisfy $(k + p)$-anonymity, and the step 2 consider the whole table to satisfy k-anonymity. The parameter p is a system parameter and it should be optimized according to results of previous trials. The pre-computation is based on the

Subset Property theorem [19]. This theorem means that each single attribute has to satisfy k-anonymity for a k-anonymous table. The mechanism uses the top-down approach and starts with a k-anonymous table as the initial table; thus the algorithm executes the following pre-computation.

1. The algorithm generalizes each attribute of a table $T(= T^q)$ to satisfy $(k+p)$-anonymity, and creates a modified table T.
2. The algorithm checks whether the whole table T satisfies k-anonymity. If the table T does not satisfy k-anonymity, then the algorithm generalizes each attribute once (one level) and returns to step 2. Note that each attribute is not generalized up to the minimum level defined by the user. If the algorithm finds no table that satisfies k-anonymity, then the algorithm outputs *failed*. Otherwise, the algorithm makes an initial anonymous table T^I and inputs it to T^G.

Top-Down Generalization. The basic steps of top-down generalization are as follows:

1. First, the algorithm de-generalizes an attribute a^n in Table T^G, which is defined as a top priority by the user, then checks the k-anonymity of the modified table T.
2. If the table satisfies k-anonymity, the algorithm calculates the score of the modified table T. If the algorithm finds no table that satisfies k-anonymity, then the algorithm outputs the table T^G and its score s. The score is computed using the score function Score(T) described in 4.2.
3. For all possible modifications of a^i $(i = 1, ..., n)$, the algorithm checks k-anonymity, then calculates the scores. Then, the algorithm selects the table that has the largest score in the possible k-anonymous table and executes step 1 again.
4. If no more tables satisfy k-anonymity, the algorithm outputs a table that has the maximum score at that point.

The basic algorithm calculates the scores of all possible k-anonymous tables. Now, we consider a more efficient method for reducing the computational costs of k-anonymity checks and score calculations. Suppose that the algorithm obtains the number $b_{w_{a^i}}$ of nodes at the same level w_{a^i} in the generalization hierarchy H_{a^i}. The number indicates the upper bound of e_{a^i} in each level. Thus, we estimate the upper bound of increase in the score to calculate $\delta_s = (b_{w_{a^i}+1} - e_{a^i})v_{a^i}$, where the current generalization level of a^i is w_{a^i}. If $s > s' + \delta_s$, the algorithm can skip de-generalization of a^i. Figure 3 shows the basic algorithm that outputs T^G and s using input parameters. The algorithm first executes pre-computation and generates an initial table T^I; then the algorithm searches a table T^G that has the highest score. The top-down approach is used for the search performed by the basic algorithm.

Input: a table T, k, p, H_{a^i}, $w^L_{a^i}$, v_{a^i}, $w^M_{a^i}$ (i=1, ..., n)
Output: T^G, s

// *Precomputation*:
Sort $(T, v_{a^1}, ..., v_{a^n})$
for $i = 1$ **to** n **do**
  $w_{a^i} \leftarrow w^L_{a^i}$
  while Check$(a^i) < k + p$ **do**
    $a^i \leftarrow$ Generalization(a^i, H_{a^i}, w_{a^i})
    $w_{a^i} \leftarrow w_{a^i} - 1$
  end while
end for
while Check$_k(T) < k$ and all $w_{a^i} > w^M_{a^i}$ **do**
  for $i = 1$ **to** n **do**
    if $w_{a^i} \geq w^M_{a^i}$ **then**
      $a^i \leftarrow$ Generalization(a^i, H_{a^i})
      $w_{a^i} \leftarrow w_{a^i} - 1$
    end if
  end for
end while
$T^I \leftarrow T$
$T^G \leftarrow T^I$
$s, s' \leftarrow$ Score(T)

if Check$_k(T) < k$ **then**
  return *failed*
  end
else
  Top-Down Generation
// *Top-Down Generation*:
while *state* $\neq$ *stop* **do**
  $T' \leftarrow T^G$
  $s' \leftarrow s$
  state $\leftarrow$ *false*
  for $i = n$ **to** 1 **do**
    $T \leftarrow T'$
    $a^i \leftarrow$ De-Generalization(a^i, H_{a^i}, w_{a^i})
    if Check$_k(T) \geq k$ and Score$(T) > s$ **then**
      $temp \leftarrow a^i, w_{a^i} + 1$
      $T^G \leftarrow T$
      $s \leftarrow$ Score(T)
      state $\leftarrow$ *true*
    end if
  end for
  if *state* $=$ *false* **then**
    state $\leftarrow$ *stop*
  end if
  $a^1, ..., a^n, v_{a^1}, ..., v_{a^n} \leftarrow temp$
end while
return T^G, s

Fig. 3. Basic Algorithm

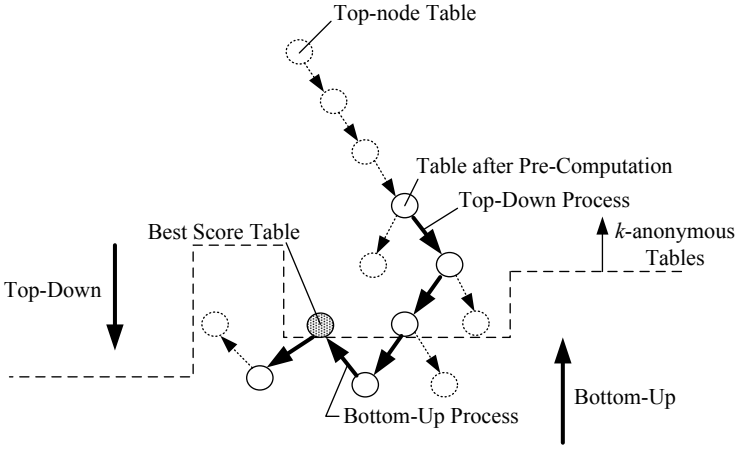


Fig. 4. Search Pass of Extended Algorithm

4.4 Extension of the Algorithm

In this section, we present an extension of the basic algorithm to find a better table that has a higher score than the table found using the basic algorithm. The extension part is executed after the execution of the basic algorithm. An

overview of the search process is shown in Figure 4. The figure is a generalization graph of a table T and the top is a root table where all attributes are level 0. The extended algorithm tries to find a table using both top-down and bottom-up approaches. The algorithm searches a high-score table on the boundary of k-anonymous tables and tables that do not satisfy k-anonymity. The extended algorithm is executed after the basic algorithm as follows:

1. After the basic algorithm, the algorithm generalizes possible a^is in the table T^G and calculates the scores of the tables. Then, the algorithm selects a top score table T^T. Note that the table T^T does not satisfy k-anonymity.
2. Next, the algorithm executes a bottom-up generalization to search tables that satisfy k-anonymity, then the algorithm chooses a table with the best score from the tables.
3. The algorithm compares the score of the table with the current score s of the table T^G and if the score is higher than s, the algorithm replaces T^G by the table T and s by the score. The algorithm executes the above steps using the new T^G. Otherwise, the algorithm stops and outputs the current generalized table T^G. The steps of top-down and bottom-up are repeated until no table has a score higher than the current best score.

Figure 6 in Appendix shows the extension algorithm in detail.

5 Analysis of Prototype System

We implemented a prototype system on a PC (Core 2 Duo 2.8GHz, 2GB Memory) and evaluated the transaction times. In this section, we show the evaluation results and a comparison.

5.1 Performance

First, we evaluated the prototype system using the Adults Data Set[6] that has 32,561 records of 14 attributes and it had also been used for a performance evaluation in previous research. The transaction times is 179 msec in the case of $k = 2$. Where $k = 10$, $k = 50$, and $k = 100$, the total transaction times are 178 msec, 137 msec and 126 msec. Next, we examined the prototype system where a huge database including many attributes was anonymized. The prototype system required 10 seconds for generating a 2-anonymization table from the Census-Income Data Set[6] that has 199,523 records of 42 attributes, and it requires 55 seconds using LandsEnd Data Set[6] that has 4,591,581 records and 8 attributes as shown in Figure 5. Thus, our prototype system is expected to generate k-anonymization tables with a feasible transaction time. Especially, where the table has records less than 100,000 and it consists of a reasonable number of attributes, the prototype system will generate a k-anonymization table of it in real-time.

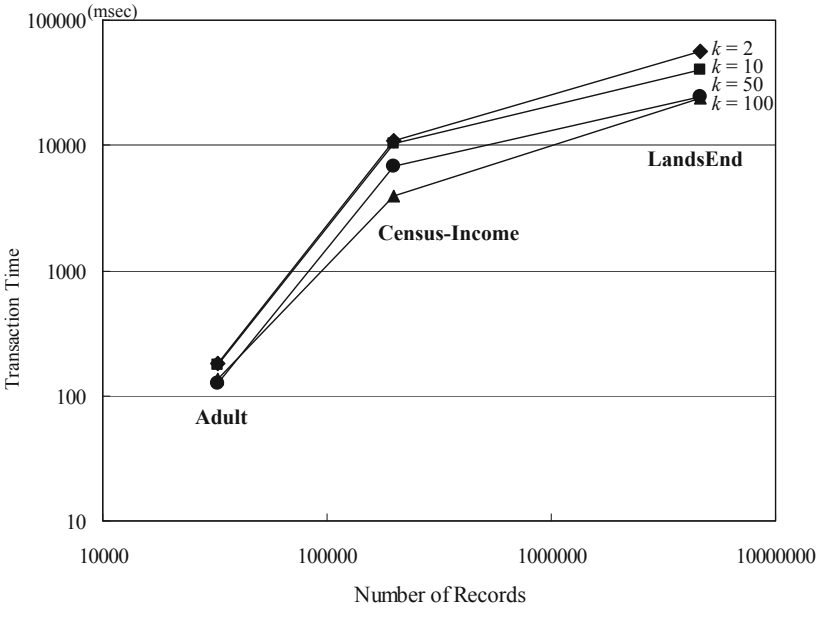


Fig. 5. Transaction Time of Prototype System

Table 1. Comparison with Top-Down Method

	No. of Records	k	No. of Check Nodes	Transaction Time	Best Score
Adult (Proposed)	32,561	2	52	226 ms	49
Adult (Top-Down)	32,561	2	598	2207 ms	49
LandsEnd (Proposed)	500,000	50	13	2083 ms	1516
LandsEnd (Top-Down)	500,000	50	540	91186 ms	1516

5.2 Comparison with Top-Down Method

We compared the prototype system under the same priority setting of all attributes with a conventional top-down method; the conventional method finds k-anonymization table simply to check all nodes from a top node using the top-down approach. Table 1 shows the comparison using two tables; the Adult Data Set and a reduced LandsEnd Data Set that has 500,000 records chosen from the original dataset. An evaluation of the top-down method using the original LandsEnd Data Set was difficult due to the huge computation; thus, we reduced the number of records and chose $k = 50$ for the evaluation. The number of checking nodes that is evaluated k-anonymity, transaction time, and the best score of k-anonymization tables with regard to the score calculation in 4.2, were compared. The number of check nodes in the top-down method is 10-40 times larger than that of the prototype system, and both methods find a table that has the best score for user's requirements. In our experiment, nine out of ten examinee fit their

desired table with the best score table generated by our system. Our mechanism would be an efficient algorithm to find a useful k-anonymization table that meets user's requirements, because it finds the best score table.

6 Conclusion

In this paper, we proposed an anonymization scheme for generating a table with k-anonymity. The scheme calculates the scores of intermediate tables based on user-defined priorities for attributes and selects the table with the highest score among them. Thus, the generated table meets a user's requirement and is employed in the services provided by users without any modification or evaluation. Our mechanism is applicable to full-domain generalization in some type of anonymity to replace the check function. We will evaluate the prototype system using several tables for several user's requirements, and consider an optimization method for system parameters and optimal score functions in our future work. Application of our concept for cell-generalization and other generalization schemes will be also considered in the future work.

References

1. Adam, N.R., Wortmann, J.C.: Security-control methods for statistical database: a comparative study. ACM Comp. Surv. 21(4), 515–556 (1989)
2. Aggarwal, C.C., Yu, P.S.: On variable constraints in privacy preserving data mining. In: Proc. of the 5th SIAM International Conference on Data Mining, pp. 115–125 (2005)
3. Aggarwal, G., Feder, T., Kenthapadi, K., Motwani, R., Panigrahy, R., Thomas, D., Zhu, A.: Anonymizing tables. In: Eiter, T., Libkin, L. (eds.) ICDT 2005. LNCS, vol. 3363, pp. 246–258. Springer, Heidelberg (2005)
4. Aggarwal, G., Feder, T., Kenthapadi, K., Motwani, R., Panigrahy, R., Thomas, D., Zhu, A.: Approximation algorithms for k-anonymity. Journal of Privacy Technology (2005)
5. Al-Fedaghi, S.S.: Balanced k-anonymity. In: Proc. of WASET, vol. 6, pp. 179–182 (2005)
6. Asuncion, A., Newman, D.J.: UCI machine learning repository (2007)
7. Bayardo, R.J., Agrawal, R.: Data privacy through optimal k-anonymity. In: Proc. of ICDE 2005, pp. 217–228 (2005)
8. Byun, J.-W., Kamra, A., Bertino, E., Li, N.: Efficient k-anonymity using clustering technique. In: Proc. of the International Conference on Database Systems for Advanced Applications, pp. 188–200 (2007)
9. Ciriani, V., De Capitani di Vimercati, S., Foresti, S., Samarati, P.: k-anonymous data mining: A survey. In: Privacy-Preserving Data Mining: Models and Algorithms. Springer, Heidelberg (2008)
10. Dalenius, T.: Finding a needle in a haystack —or identifying anonymous census record. Journal of Official Statistics 2(3), 329–336 (1986)
11. Duncan, G., Lambert, D.: The risk of disclosure for microdata. J. Buisiness & Economic Statistics 7, 207–217 (1989)
12. Dwork, C.: Differential privacy. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. LNCS, vol. 4052, pp. 1–12. Springer, Heidelberg (2006)

13. Dwork, C.: Differential privacy: A survey of results. In: Agrawal, M., Du, D.-Z., Duan, Z., Li, A. (eds.) TAMC 2008. LNCS, vol. 4978, pp. 1–19. Springer, Heidelberg (2008)
14. Dwork, C., Kenthapadi, K., McSherry, F., Mironov, I., Naor, M.: Our data, ourselves: Privacy via distributed noise generation. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 486–503. Springer, Heidelberg (2006)
15. Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 265–284. Springer, Heidelberg (2006)
16. Fienberg, S.E., McIntyre, J.: Data swapping: Variations on a theme by dalenius and reiss. In: Domingo-Ferrer, J., Torra, V. (eds.) PSD 2004. LNCS, vol. 3050, pp. 14–29. Springer, Heidelberg (2004)
17. Iyengar, V.S.: Transforming data to satisfy privacy constraints. In: Proc. of ACM SIGKDD 2002, pp. 279–288. ACM, New York (2002)
18. Kiyomoto, S., Martin, K.M.: Towards a common notion of privacy leakage on public database. In: Proc. of BWCCA 2010 (2010) (to appear)
19. LeFevre, K., DeWitt, D.J., Ramakrishnan, R.: Incognito: Efficient full-domain k-anonymity. In: Proc. of SIGMOD 2005, pp. 49–60 (2005)
20. LeFevre, K., DeWitt, D.J., Ramakrishnan, R.: Mondrian multidimensional k-anonymity. In: Proc. of the 22nd International Conference on Data Engineering (ICDE 2006), pp. 25–35. IEEE, Los Alamitos (2006)
21. LeFevre, K., DeWitt, D.J., Ramakrishnan, R.: Workload-aware anonymization. In: Proc. ACM SIGKDD 2006, pp. 277–286. ACM, New York (2006)
22. Lin, J.-L., Wei, M.-C.: An efficient clustering method for k-anonymization. In: Proc. of the 2008 International Workshop on Privacy and Anonymity in Information Society (PAIS 2008), pp. 46–50. ACM, New York (2008)
23. Loukides, G., Tziatzios, A., Shao, J.: Towards preference-constrained k-anonymisation, pp. 231–245 (2009)
24. Machanavajjhala, A., Gehrke, J., Kifer, D.: l-diversity: Privacy beyond k-anonymity. In: Proc. of ICDE 2006, pp. 24–35 (2006)
25. Machanavajjhala, A., Gehrke, J., Kifer, D.: t-closeness: Privacy beyond k-anonymity and l-diversity. In: Proc. of ICDE 2007, pp. 106–115 (2007)
26. Meyerson, A., Williams, R.: On the complexity of optimal k-anonymity. In: Proc. of PODS 2004, pp. 223–228 (2004)
27. Miller, J., Campan, A., Truta, T.M.: Constrained k-anonymity: Privacy with generalization boundaries. In: Proc. of the Practical Preserving Data Mining Workshop, P3DM 2008 (2008)
28. Samarati, P.: Protecting respondents' identities in microdata release. IEEE Trans. on Knowledge and Data Engineering 13(6), 1010–1027 (2001)
29. Samarati, P., Sweeney, L.: Generalizing data to provide anonymity when disclosing information. In: Proc. of the 17th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS 1998), p. 188 (1998)
30. Sun, X., Wang, H., Li, J., Truta, T.M., Li, P.: (p^+, α)-sensitive k-anonymity: a new enhanced privacy protection model. In: Proc. of CIT 2008, pp. 59–64 (2008)
31. Sweeney, L.: Achieving k-anonymity privacy protection using generalization and suppression. J. Uncertainty, Fuzziness, and Knowledge-Base Systems 10(5), 571–588 (2002)
32. Truta, T.M., Campan, A.: K-anonymization incremental maintenance and optimization techniques. In: Proceedings of the 2007 ACM Symposium on Applied Computing (SAC 2007), pp. 380–387. ACM, New York (2007)
33. Truta, T.M., Vinay, B.: Privacy protection: p-sensitive k-anonymity property. In: Proc. of ICDE 2006, pp. 94–103 (2006)

34. Willenborg, L., de Waal, T.: Elements of Statistical Disclosure Control. LNS, vol. 155. Springer, Heidelberg (2001)
35. Winkler, W.E.: Masking and re-identification methods for public-use microdata: Overview and research problems. In: Domingo-Ferrer, J., Torra, V. (eds.) PSD 2004. LNCS, vol. 3050, pp. 231–246. Springer, Heidelberg (2004)
36. Wong, R.C.-W., Li, J., Fu, A.W.-C., Wang, K.: (α, k)-anonymity: an enhanced k-anonymity model for privacy preserving data publishing. In: Proc. of ACM SIGKDD 2006, pp. 754–759 (2006)
37. Xiao, X., Tao, Y.: Personalized privacy preservation. In: Proc. of SIGMOD 2006, pp. 229–240. ACM, New York (2006)
38. Xu, J., Wang, W., Pei, J., Wang, X., Shi, B., Fu, A.W.-C.: Utility-based anonymization for privacy preservation with less information loss. SIGKDD Explor. Newsl. 8(2), 21–30 (2006)
39. Xu, J., Wang, W., Pei, J., Wang, X., Shi, B., Fu, A.W.-C.: Utility-based anonymization using local recoding. In: Proc. of ACM SIGKDD 2006, pp. 785–790. ACM, New York (2006)
40. Zhu, H., Ye, X.: Achieving k-anonymity via a density-based clustering method. In: Dong, G., Lin, X., Wang, W., Yang, Y., Yu, J.X. (eds.) APWeb/WAIM 2007. LNCS, vol. 4505, pp. 745–752. Springer, Heidelberg (2007)

A Details of Extension Algorithms

Figure 6 shows the detailed algorithm that outputs T^G and s using input from the basic algorithm. The algorithm repeats two sub-functions until no table has a score higher than the current best score.

Input: a table T^G, k, s, H_{a^i}, $w^L_{a^i}$, v_{a^i}, $w^M_{a^i}$ (i=1, ..., n)
Output: T^G, s

repeat
// *2nd Top-Down*:
$T' \leftarrow T^G$
$s' \leftarrow s$
$state \leftarrow false$
for $i = n$ **to** 1 **do**
  $T \leftarrow T'$
  $a^i \leftarrow$ De-Generalization(a^i, H_{a^i}, w_{a^i})
  if Score$(T) > s'$ **then**
    $temp \leftarrow a^i, w_{a^i} + 1$
    $T^T \leftarrow T$
    $s^T \leftarrow$ Score(T)
    $state \leftarrow true$
  end if
end for
if $state = false$ **then**
  $state \leftarrow stop$
  return T^G, s
else
// *Bottom-Up*:
while Check$_k(T) < k$ **do**
$T' \leftarrow T^T$
$state \leftarrow false$
$a^1, ..., a^n, v_{a^1}, ..., v_{a^n} \leftarrow temp$
  for $i = 1$ **to** n **do**
    $T \leftarrow T'$
    if $w_{a^i} > w^M_{a^i}$ **then**
      $a^i \leftarrow$ Generalization(a^i, H_{a^i}, w_{a^i})
    end if
    if Check$_k(T) \geq k$ and Score$(T) > s$ **then**
      $temp \leftarrow a^i, w_{a^i} - 1$
      $T^G \leftarrow T$
      $s \leftarrow$ Score(T)
      $state \leftarrow true$
    end if
  end for
  if $state = true$ **then**
    $a^1, ..., a^n, v_{a^1}, ..., v_{a^n} \leftarrow temp$
  else
  for $i = 1$ **to** n **do**
    $T \leftarrow T'$
    if $w_{a^i} > w^M_{a^i}$ **then**
      $a^i \leftarrow$ Generalization(a^i, H_{a^i}, w_{a^i})
    end if
    if Score$(T) > s'$ **then**
      $T^T \leftarrow T$
      $temp \leftarrow a^i, w_{a^i} - 1$
      $s' \leftarrow$ Score(T)
      $state \leftarrow true$
    end if
  end for
  if $state = false$ **then**
    $state \leftarrow stop$
    return T^G, s
  end if
end while
until $state = stop$

Fig. 6. Extension of the Algorithm

FAANST: Fast Anonymizing Algorithm for Numerical Streaming DaTa

Hessam Zakerzadeh and Sylvia L. Osborn

Department of Computer Science
The University of Western Ontario
hzakerza@uwo.ca, sylvia@csd.uwo.ca

Abstract. Streaming data is widely used in today's world. Data comes from different sources in streams, and must be processed online and with minimum delay. These data streams usually contain confidential data such as customers' purchase information, and need to be mined in order to reveal other useful information like customers' purchase patterns. Privacy preservation throughout these processes plays a crucial role. K-anonymity is a well-known technique for preserving privacy. The principle issues in k-anonymity are data loss and running time. Although some of the existing k-anonymity techniques are able to generate anonymized data with acceptable data loss, their main drawback is that they are very time consuming, and are not applicable in a streaming context since streaming data is usually very sensitive to delay, and needs to be processed quite fast. In this paper, we propose a cluster-based k-anonymity algorithm called FAANST (Fast Anonymizing Algorithm for Numerical Streaming daTa) which can anonymize numerical streaming data quite fast, while providing an admissible data loss. We also show that FAANST can be easily extended to support data streams consisting of categorical values as well as numerical values.

1 Introduction

Streaming data is being widely used in today's world. Data from different sources comes in streams, and is required to be processed online and with minimum delay. Examples of applications using data streams are financial applications, network monitoring applications, security applications, telecommunications data management applications, web applications, manufacturing applications, and others. Streams from these applications need to be mined in order to obtain important information. As a case in point, online trading companies such as *Questrade.com* receive and record tens of thousands of online bids issued by their customers everyday. Suppose the bids made by bidders constitute a transaction stream having the schema *S(firstname, lastname, gender, zipcode, age, company-name, quantity)*. To find statistical information for each company's stock, and publish it in a real-time manner, a mining algorithm processes the data stream produced by these transactions. Low delay and customers' privacy are the two most significant issues in this processing. To cope with the delay issue, fast algorithms should be

J. Garcia-Alfaro et al. (Eds.): DPM 2010 and SETOP 2010, LNCS 6514, pp. 36–50, 2011.

used, and in order to protect the customers' privacy, the attributes that explicitly identify the customers (such as *firstname*, and *lastname*) should be removed from the data stream. However, in most cases, it may still be possible to reveal the customers' identity by a linking attack. Linking attacks occur when released data can be linked with publicly available data in order to reveal individuals' identification. According to [18] more than 85% of the United States' population can be uniquely identified by the combination of gender, zipcode, and date of birth.

Privacy preservation in sharing data streams, which can contain confidential and secret information of individuals, between data holders who are individuals and those running queries on these data streams has received a lot of attention in both research and applications. Individuals do not desire their identification and sensitive information to be revealed to those who are running queries. To achieve this goal, a well-known privacy-preserving technique for anonymizing data called k-anonymity can be used. The main goal in k-anonymity is to prevent linking attacks.

Consider the data holder's dataset to be a private dataset, *PDS,* in which each tuple refers to a different entity (individual, organization, etc). The data holder constructs a new dataset, which is an anonymous version of *PDS* to release. All explicit identifiers should be either suppressed or encrypted in the anonymous dataset. Attributes on which linking can be enforced are called Quasi-Identifiers. Quasi-Identifiers are defined as follows:

Definition 1. *(Quasi-Identifiers): let DS* $(A_1,\ldots,A_n)$ *be a dataset. A quasi-identifier of DS is a set of attributes* $\{A_i,\ldots,A_j\} \subseteq \{A_1,\ldots,A_n\}$ *whose release must be controlled[16,15].*

Quasi-identifiers can be used to link an anonymous dataset with publicly available data in order to disclose the identity of individuals. The goal is to release the information in the dataset while the privacy of the individuals is preserved through anonymity. The anonymity constraint requires released information not to be distinguishable from at least a given number of k individuals. This is called the k-anonymity constraint.

Definition 2. *(K-Anonymity): let DS*$(A_1,\ldots A_n)$ *be a dataset and QI be the quasi-identifiers associated with it. DS is said to satisfy k-anonymity wrt QI if and only if each sequence of values in DS[QI] appears with at least k occurrences in DS[QI][16,15].*

Anonymization can be achieved through two techniques: Generalization and Suppression. In Generalization, the values of quasi-identifiers are replaced with more general values. For example, suppose country is one of the quasi-identifiers in a dataset. Generalization says that a value should be replaced by a more general value, so that the country would be replaced by the continent where the country is located. Suppression, on the other hand, means removing data from the dataset so that it is not released at all. Generalization and Suppression are complementary techniques. Table 1a shows a portion of the customers' dataset

Table 1. Customers' data

(a) Original customers' data

firstname	lastname	gender	zipcode	age	company-name	quantity
Anna	Mackay	Female	20433	27	ECU Mining corp.	1200
Bob	Miller	Male	20436	30	M Split corp.	180
Carol	King	Female	20456	45	MAG Silver corp.	2500
Daisy	Herrera	Female	20489	53	Warnex Inc.	10500
Alicia	Bartlett	Female	20411	41	M Split corp.	300

(b) 2-Anonymized version of customers' data

gender	zipcode	age	company-name	quantity
NA	2043*	[25-35]	ECU Silver corp.	1200
NA	2043*	[25-35]	M Split corp.	180
Female	204**	[40-55]	MAG Silver corp.	2500
Female	204**	[40-55]	Warnex Inc.	10500
Female	204**	[40-55]	M Split corp.	300

for a trading company in which *firstname* and *lastname* are identifying attributes, and *gender, zipcode,* and *age* are quasi-identifiers. Table 1b is a 2-anonymized version of Table 1a.

The principal issues in generalization and suppression are data loss and time complexity. In achieving k-anonymity, data can be either expressed in a more general form or suppressed, which results in data loss. We always intend to minimize the data loss resulting from Generalization and Suppression in order to preserve as much as possible the quality of the released data. If we generalize or suppress data more than what is required, the data will not be useful anymore. The other issue is time complexity. Computing optimized k-anonymity is NP-hard [12], and leads to significant computational challenges. So, most of the existing techniques cannot be used in a streaming context since as mentioned above, delay is an important factor in streaming data, and the optimal techniques are quite slow. A naive approach for anonymizing streaming data is to buffer the tuples, and then anonymize them. This approach fails because first it implies a huge storage overhead. Second, this method causes delay between data arrival and anonymized data because data should be buffered, and it causes delay which is not desirable in streaming data. In this paper, we introduce a new cluster-based algorithm for anonymizing numerical data streams using window processing, and it will be shown that this algorithm is capable to anonymize the streaming data quite fast while having low data loss.

The remainder of this paper is organized as follows. In Sect. 2, we review the existing techniques for anonymizing both static and streaming data. Sect. 3 encompasses a detailed description of our newly-proposed algorithm as well as its time complexity analysis. We present our experimental results in Sect. 4, and finally Sect. 5 concludes the paper.

2 Related Work

Generally, k-anonymity approaches fall into two main categories: Hierarchy-Based Generalization and Hierarchy-Free Generalization. In Hierarchy-Based Generalization, the domain of each attribute should be specified through a hierarchy (taxonomy tree). This hierarchy is called a Domain Generalization Hierarchy or *DGH*. *DGH* consists of values which each attribute can accept [16]. From *DGH*, a Value Generalization Hierarchy (*VGH*) can be constructed. *DGH* and *VGH* are the main components in Hierarchy-Based Generalization. For hierarchy-based algorithms, the *DGH* and *VGH* of each attribute must be specified by a data analyst before starting to generalize.

Datafly [17] and μ-Argus [5] were the first two real-world systems using hierarchy-based generalization. These two systems use heuristics to make approximation for k-anonymity, but they do not always yield optimal results. In a technique proposed by Samarati and Sweeney [15], a lattice on distance vector is created, and the process of selecting the minimal k-anonymized data is conducted based on this lattice. The Incognito algorithm [7] generates the set of all possible k-anonymous generalizations of table T which are to be anonymized, with an optional tuple suppression threshold. Based on the Subset Property which the authors have defined, this algorithm starts to see if T preserves k-anonymity with respect to single-attribute subsets of quasi-identifiers. Then, it checks k-anonymity with respect to double-attribute subsets of quasi-identifiers, and so on. Other approaches such as [4,20,10] have also introduced other hierarchy-based techniques for achieving k-anonymity. Unlike hierarchy-based generalization which needs a user-defined domain hierarchy, the majority of hierarchy-free generalization solutions can be done automatically. This type of solution takes advantage of clustering and partitioning to produce a generalized result. Bayardo et al. [2], has presented a new approach for exploring the space of possible anonymizations. Their approach starts with a fully generalized dataset, and iteratively specializes the dataset into one that is minimally k-anonymous. Multidimensional k-anonymity [8] is another hierarchy-free technique for generalization. It proposed a greedy algorithm to partition the space which has been created by attribute domains, multidimensionally. The previously mentioned k-anonymization techniques have been designed for static data, and are quite slow. They are not capable of addressing the requirements of continuous, transient, and unbounded data streams. CASTLE [3] is a cluster-based scheme that anonymizes data streams on-the-fly, and ensures the freshness of anonymized data by satisfying specified delay constraints. It uses a count-based window for processing the streaming data. Bin Zhou et al. [22], have also developed a method in three steps for generalization in data streams. In the first step, they have proposed a randomized algorithm which makes decisions about publishing based on the information loss of equivalence classes, but does not consider the distribution of the data stream to be published. In the second step, they incorporate the distribution of the data stream into the decision making procedure. In the last step, to further reduce the information loss, they examine the potential of publishing with future tuples for every tuple arriving. SKY [9] and SWAF [19] are

two other attempts at anonymizing streaming data which take a specialization tree as input, and based on the specialization tree, generate anonymized data. The problem with [3,22] is that they are not fast enough, and [9,19] need a specialization tree even for numerical values. To the best of our knowledge, these four algorithms are the only work on k-anonymity for streaming data.

3 FAANST: Fast Anonymizing Algorithm for Numerical Streaming DaTa

As mentioned earlier, delay in a streaming context is of high importance. Any processing on streaming data, including anonymization, should have as low a delay as possible. FAANST is a cluster-based anonymizing algorithm for streaming data consisting of numerical values, but as will be mentioned later, it can be easily extended to support categorical values as well as numerical values. In this section, before digging into the details of the proposed algorithm, we will explain the underlying concepts for our algorithm including the information loss metric and the clustering algorithm used in it. After that, we analyse the time complexity of this algorithm.

FAANST takes three parameters to operate: k, *MU*, and *DELTA*. k is the common parameter among all k-anonymity algorithms, and determines the degree of anonymity. *MU* specifies the size of the processing window (the processing window is the portion of the stream under consideration). Window processing in our approach is a combination of two window processing techniques: Landmark and Count-Based window processing techniques[1]. In other words, the starting bound of the window in our algorithm is fixed, and is when the algorithm starts to run; the ending bound slides as time elapses (like Landmark window processing), but the storage capacity is *MU* tuples (like Count-Based window processing). When the number of tuples in the processing window reaches *MU*, we run the first round of the clustering algorithm and some of the tuples are output; then the window can slide again in order to accumulate more tuples. In this first case, it is possible that some tuples which are not among the N most recent tuples exist in the window, as there is no guarantee of outputting only old tuples. In future work we plan to consider how to better deal with old tuples. Some of the clusters output will be of better quality than others. *DELTA* is used to decide this. Clusters whose data loss is less than *DELTA* and which contain more than k tuples are called *accepted* clusters and are kept to be reused in the next rounds. By reused we mean that later tuples whose values fall into a reusable cluster are output to that cluster. All clusters output satisfy k-anonymity, but only those with acceptable data loss are reused.

3.1 Information Loss Metrics

The information loss metric is a critical issue in evaluating k-anonymity algorithms, and can be computed at once for anonymizing static data. In contrast,

[1] For more information on window processing please refer to [14].

in a streaming context, it should be calculated in an incremental manner, and the final data loss is the sum of partial data loss. The information loss metric proposed by [6] has been adopted by many anonymizing algorithms, and we adopt it as well. Suppose the domain of the attribute A_i is $[v_{min}, v_{max}]$, and value $v_i \in [v_{min}, v_{max}]$ falls into cluster $[c_{min}, c_{max}]$ after anonymization. Then, the information loss for this value (cluster) is:

$$infoloss(v_i) = \frac{c_{max} - c_{min}}{v_{max} - v_{min}}$$

Example 1. Suppose age is a numerical attribute with domain range *[0,100]*. If the clustering algorithm partitions this range into four sub-partitions {*[0,15]*, *[15,30]*, *[30, 60]*, *[60,100]*}, the information loss of the generalized value for 35, which is *[30,60]*, is $\frac{60-30}{100-0} = 0.3$.

The information loss of a generalized tuple $g = (v_1, v_2, ..., v_n)$ is calculated by averaging the information loss of all attributes (dimensions) as below:

$$infoloss(g) = \frac{1}{n} \sum_{i=1}^{n} infoloss(v_i)$$

The total information loss of a data stream is simply calculated by averaging the information loss of all objects in it. Here is an illustrative example of calculating the total information loss using this incremental metric.

Example 2. Suppose our schema is (*Age*, *Number-of-children*) in which the domains of *Age* and *Number-of-children* are *[0-100]* and *[0-15]*, respectively. We have two objects with values *(28, 2)* and *(51, 4)*. Imagine these two objects are generalized as (*[15,30]*,*[0,3]*) and (*[30,60]*,*[4,6]*). Thus, the total data loss will be $\frac{1}{2}(infoloss(object_1) + infoloss(object_2)) = \frac{1}{2}(\frac{1}{2} \times (\frac{30-15}{100-0} + \frac{3-0}{15-0}) + \frac{1}{2} \times (\frac{60-30}{100-0} + \frac{6-4}{15-0})) \simeq 0.2$.

To make data loss comparisons with other approaches easier, we have adopted this commonly-used data loss metric.

3.2 Pseudocode

Clustering is the most important part of FAANST. We have used K-means[2] [11], which is one of the most popular and widely-used partitioning algorithms, as our partitioning algorithm. This algorithm is quite simple and fast, and because of these two properties we have used it in FAANST for clustering data streams requiring fast processing.

We describe FAANST in two parts (a round corresponds to what happens when the storage is filled with data from the stream): (1) the first round that the algorithm runs, and (2) when at least one round has been executed.

For the first round, when MU tuples arrive (meaning that the window is full), the algorithm partitions MU tuples into k' clusters using k-means[3]. If the number of tuples in a cluster c_i ($1 \leq i \leq k'$)is greater than or equal to k, all tuples

[2] We used our own implementation of k-means.

[3] We refer to the number of clusters in k-means as k'in order to prevent confusion with k in k-anonymity.

in c_{i} are output, and provided that data loss of c_{i} is less than *DELTA*, c_{i} will be added to the list of accepted clusters which will be reused in subsequent rounds to output tuples. In other words, tuples in clusters having condition $clusterSize \geq \mathrm{k}$ are output in this phase, and those clusters which have two conditions $clusterSize \geq \mathrm{k}$ and $infoloss \leq DELTA$ are stored in the list of accepted clusters. After outputting these tuples in the first round, the algorithm should wait in order to fill its memory again before proceeding.

When the number of tuples in the window reaches *MU* again, in the second round to the end, it acts as follows: tuples falling into one of the accepted clusters found in previous rounds are output to that cluster without considering the number of tuples in that cluster because we know that at least k tuples have already been output in this cluster in previous rounds, and k-anonymity will not be violated. Then, it partitions the remaining tuples into $k^{'}$ clusters, and outputs tuples falling in clusters with more than k tuples. Again, clusters having more than k tuples and whose data loss is less than *DELTA* will be added to the list of accepted clusters. Afterwards, it waits for the window to accumulate *MU* tuples once again, and goes to the next round.

In the end, when no more tuples come, and the number of tuples in the current window is less than *MU*, the algorithm checks all remaining tuples one by one to see if they fit into one of the accepted clusters or not. If so, they will be output to that cluster. After this phase, for the remaining tuples, while the number of remaining tuples is larger than k, the algorithm iteratively clusters them into $k^{'}$ clusters, and outputs tuples falling into clusters having more than k tuples. When fewer than k tuples remain, they are suppressed. This step ensures us that the maximum number of suppressed tuples is k.

Here we exemplify our algorithm in order to better understand it. Suppose a data stream has two quasi-identifier attributes whose domains are [0-100] and [0-10], age and number-of-children. *MU*, k, and *DELTA* are set to 2000, 50, and 0.4, respectively. In the first round, when 2000 (*MU*) tuples come, we partition them into (2000/50)=40 clusters using k-means. Suppose only two clusters contain more than 50 (k) tuples, ([20-30],[0-2]) with 62 tuples and data loss less than *DELTA* and ([30-75],[3-9]) with 110 tuples and data loss greater *DELTA*. Thus, we output the tuples falling into these two clusters, 62 tuples with value ([20-30],[0-2]) and 110 tuples with value ([30-75],[3-9]), and keep the other tuples in the system to be output in the next rounds. We also store only the cluster ([20-30],[0-2]) for future reuse since it has data loss is less than *DELTA* and has more than k tuples, and forget the rest of the clusters. What we mean by storing a cluster is simply keep the cluster's range, not its tuples. In the next rounds, first each tuple is checked for inclusion in one of the stored clusters. If it falls into one of them, this tuple is generalized by the range of the cluster which contains it, and if not, this tuple remains in the system. Afterwards, we partition the remaining tuples into (the number of tuples in the current processing window/k) clusters, and repeat exactly the first round's process. In the end, if no more tuples come, the remaining tuples which do not fit into any of the stored clusters, should be suppressed.

It is worth mentioning what outputting and suppressing mean in our algorithm. Recall Example 2 for calculating information loss. If tuple (*28, 2*) falls into cluster ([*15,30*],[*0,3*]), we output ([*15,30*],[*0,3*]), which is the cluster range, instead of this tuple. Should (*28, 2*) need to be suppressed, we output tuple ([*0,100*],[*0,15*]), which is the most generalized data, rather than this tuple.

The fundamental question arising here is how we can calculate the value of $k^{'}$ in each round. Suppose *sizeOfCurrentWindow* denotes the number of tuples in the current window, and the degree of anonymity is k. Since this algorithm is for streaming context which is sensitive to delay, we do not desire to keep tuples for a long time in the system. So, we choose $k^{'}$ such that it ensures us of outputting tuples in at least one cluster in each round of clustering. That is, we desire to output some tuples in each round in order to reduce delay. Selecting small values for $k^{'}$ can guarantee this requirement (i.e., if we select $k^{'}$=*1*, which means only one cluster exists, this causes all tuples to be output since they all fall into one cluster whose size is *sizeOfCurrentWindow*). On the other hand, the quality of the output data is of great importance. If we generalize the data too much, the output data will not be useful. This requirement implies that the value of $k^{'}$ should be as large as possible since having a great number of clusters leads to better quality for each cluster. To accomplish these two requirements (outputting tuples in at least one cluster in each round and having high quality output data), we select $k^{'}$ equal to (*sizeOfCurrentWindow*/k)[4]. Based on the Pigeonhole principle, this value is the greatest value for $k^{'}$ that ensures us that at least one cluster with k tuples exists and can be output. Algorithm 1 shows the pseudocode of FAANST.

Algorithm 1. FAANST *(MU,k,DELTA)*

```
1: while (More tuples come) do
2:     read a tuple and add it to currentWindow
3:     if (size of the currentWindow = MU) then
4:         comment: re-use part
5:         foreach(tuple t in the currentWindow)
6:         if (t falls into one or more of the stored clusters) then
7:             output t with cluster having the lowest data loss and in which
               t falls.
8:         end if
9:         end foreach
10:        comment: clustering part
11:        if (size of currentWindow ≠ 0) then
```

[4] There is only one case in which this requirement may not be satisfied, and it occurs when less than k tuples exist in the window when we are about to cluster them. So, selecting $k^{'} = \frac{sizeOfCurrentWindow}{k}$ will not incur outputting tuples in at least one cluster.

partition $currentWindow$ into ($|currentWindow|/k$) clusters using clustering algorithm (e.g. k-means).
output tuples falling into cluster c_i which has more than k tuples, and store c_i if $infoloss(c_i) < DELTA$.
end if
end if
end while
comment: when no more tuples come
foreach(tuple t in the $currentWindow$)
if (t falls into one or more of the stored clusters) **then**
output t with cluster having the lowest data loss and in which t falls.
end if
end foreach
while ($|currentWindow| > k$) **do**
partition $currentWindow$ into ($|currentWindow|/k$) clusters using clustering algorithm.
output tuples falling into cluster c_i which has more than k tuples.
end while
Suppress the remaining tuples.

The time complexity of FAANST is $O(\frac{i^2 \times MU^2}{k})$ where i is the number of rounds of the algorithm[5]. This value depends on values of $|S_{in}|$[6], $DELTA$, MU, and k, and in our experiments varies between [6 and 60] with ($|S_{in}| \in \{10922, 30162, \}$, $0.5 \leq DELTA \leq 1$, $500 \leq MU \leq 2500$, and 10 $\leq$k$\leq$ 300).

4 Experimental Results

In this section, we present the results of several experiments we have done on both synthetic and real numerical data for our proposed algorithm, and compare it with CASTLE which is the most similar algorithm to FAANST. We have implemented FAANST in C++ using Visual Studio 2005, and all the experiments have been performed on a machine with a 2.4 GHz Intel Core 2 DUO processor and 3GB RAM. Our aims in conducting these experiments are studying the effects of anonymity degree (k), window size (MU), and quality of clusters ($DELTA$) on data loss, run-time, and the number of suppressed tuples.

In parts 4.1 and 4.2, we show the results of running FAANST on two different datasets from the UCI Machine Learning Repository [1] . The first experiment has been conducted on the synthetic numeric dataset Pen-Based Recognition of Handwritten Digits Dataset, and the second experiment has been done on numeric attributes of real dataset Adult which consists of census data, which have been de facto datasets for k-anonymity algorithms. The results of these

[5] Complete analysis is available in [21].
[6] The total number of tuples.

experiments[7] show that our proposed algorithm outperforms CASTLE in terms of data loss, running time, and the number of suppressed tuples. We have done each experiment five times, and averaged the results.

4.1 Experiment 1

The Pen-Based Recognition of Handwritten Digits Datasets contains 10992 sample tuples from 44 writers. Each writer has been asked to write random digits in a sensitive tablet with dimensions of 500 by 500 pixels. Then, 8 sample points have been taken from each written digit in the form of *(x,y),* and transformed into the range of 0 to 100. Thus, the number of quasi-identifiers in this dataset is 16. Diagrams below show the effects of *MU*, k, and *DELTA* on data loss, runtime, and number of suppressed tuples. We have varied *MU* from 500 to 2500, k from 50 to 300, and *DELTA* from 0.5 to 1.

As Fig. 1a shows, data loss increases when the anonymity degree becomes larger, but by enlarging the window size, it is possible to counteract the effect of increase in the anonymity degree a little. This figure also depicts that *DELTA* has a minor effect on the amount of data loss.

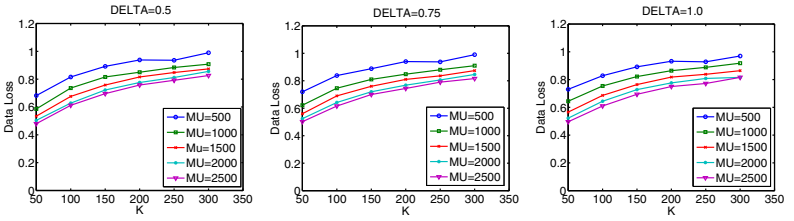
Based on Fig. 1b, when making the anonymity degree larger, running time decreases, and is almost higher for larger windows than smaller ones. Besides, larger *DELTA* reduces the running time. Peaks in these graphs are mainly because of randomness in the first phase of k-means leading to improper selection of centroid points in the initialization phase of k-means which results in more rounds, varying between 6 and 60 in our experiments, and longer running time. According to the time complexity, although the running time should be proportional to MU^2, looking horizontally across Fig. 1b does not demonstrate this. That is because the runtime also depends on i^2, the number of rounds.

The maximum number of suppressed tuples is k. Figure 1c shows that the number of suppressed tuples for larger anonymity degrees is usually higher than smaller anonymity degrees. The number of suppressed tuples in each diagram is the average of the number of suppressed tuples per $MU = 500, 1000, 1500, 2000$, and 2500.

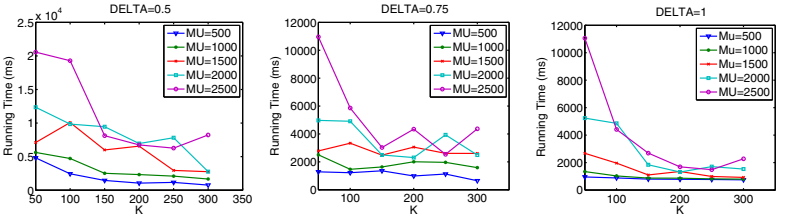
Figure 2 shows the results of comparison of three metrics data loss, running time, and number of suppressed tuples with CASTLE. In these comparisons, we have limited *MU* to 500, 1500, and 2500 and *DELTA* to 0.5 and 1. As can be seen from these graphs, FAANST, with one exception outperforms CASTLE in terms of data loss and running time. Furthermore, the number of suppressed tuples for FAANST with *DELTA*=1 is always lower than CASTLE. CASTLE does not have any guarantee on the number of suppressed tuples, but FAANST guarantees the number of suppressed tuples to be not more than k.

Based on the experiments on this dataset, smaller *DELTA* improves data loss very slightly, but increases the running time and number of suppressed tuples. So, in the next experiment, we only consider *DELTA*=1.

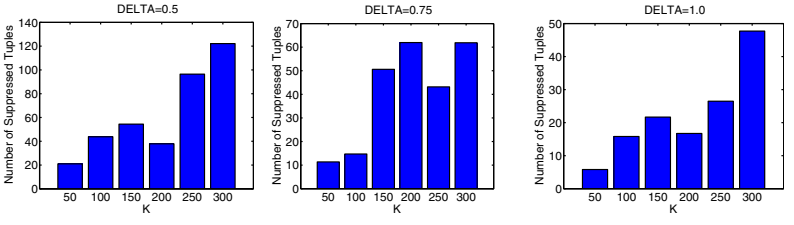
[7] These comparisons have been done using source code which the authors of [3] kindly provided for us.



(a) Data loss vs. k and MU on Dataset 1, Numeric data



(b) Running time vs. k and MU on Dataset 1, Numeric data



(c) Average number of suppressed tuples per MU = 500,1000,1500,2000, and 2500 vs. k on Dataset 1, Numeric data

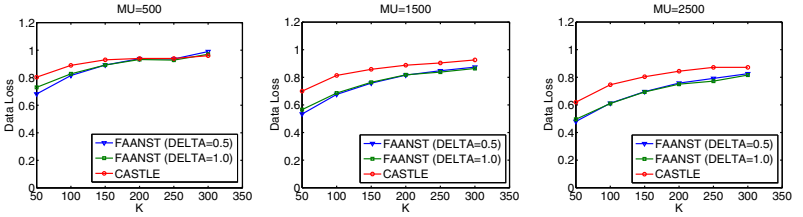
Fig. 1. Results of running FAANST on Pen-Based Recognition of Handwritten Digits Dataset

4.2 Experiment 2

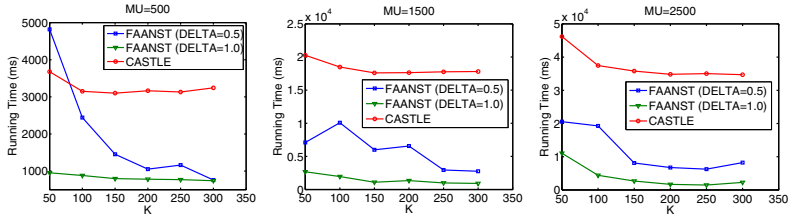
The second experiment has been conducted on a real dataset, Adult, which consists of US census data, and has become the de facto dataset for k-anonymity algorithms. We run our experiments only on the numerical attributes of this dataset. The Adult dataset encompasses 48842 instances in total, but we have eliminated instances with missing values. So, our final dataset has 30162 instances. Each instance includes numerical attributes: *age*, *final-weight*, *education-number*, *capital-gain*, *capital-loss*, and *hours-per-week*. Table 2 shows the domain of each numerical attribute in the Adult dataset.

Similarly to the previous set of experiments, we have varied MU from 500 to 2500 and k from 50 to 300, and measure the effect of these changes on data loss, running time, and number of suppressed tuples. *DELTA* is also fixed and set to 1 for all the experiments on this dataset. Figure 3 shows these results.

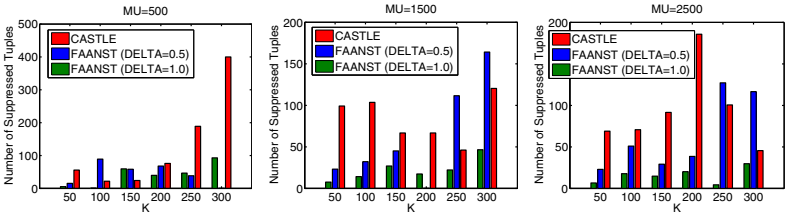
The results of comparison with CASTLE are available in Fig. 4. We vary k from 50 to 400 in these experiments to show the difference better. Although



(a) Data loss comparison with CASTLE on Dataset 1, Numeric data



(b) Running time comparison with CASTLE on Dataset 1, Numeric data



(c) Number of suppressed tuples comparison with CASTLE on Dataset 1, Numeric data

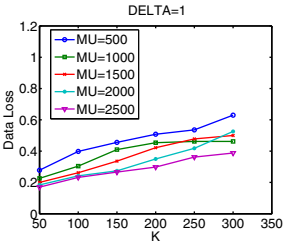
Fig. 2. Results of comparison with CASTLE on Pen-Based Recognition of Handwritten Digits Dataset

Table 2. Domain of Adult's numerical attributes dataset

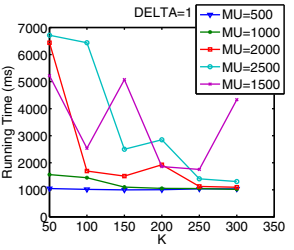
	Age	Final -weight	Education-number	Capital-gain	Capital-loss	Hours-per-week
Min	17	13492	1	0	0	1
Max	90	1490400	16	99999	4356	99

the number of suppressed tuples of FAANST in these experiments is not always lower than CASTLE, FAANST has lower data loss and running time on the Adult dataset.

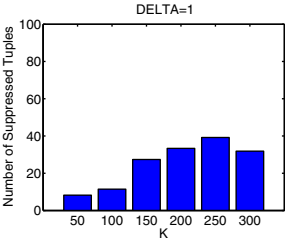
These graphs are consistent with the results of the previous dataset. Data loss has a direct relation with k and an inverse relation with *MU*. In contrast, running time has a direct relation with *MU* and inverse relation with k, and the number of suppressed tuples cannot exceed k. Again, peaks in running time graph are because of unsuitable selection of initial centroids in k-means which results in higher running time.



(a) Data loss vs. k and *MU* on Dataset 2 for *DELTA*=1, Numeric data

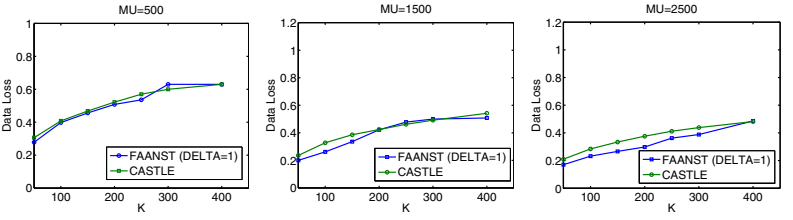


(b) Running time vs. k and *MU* on Dataset 2 for *DELTA*=1, Numeric data

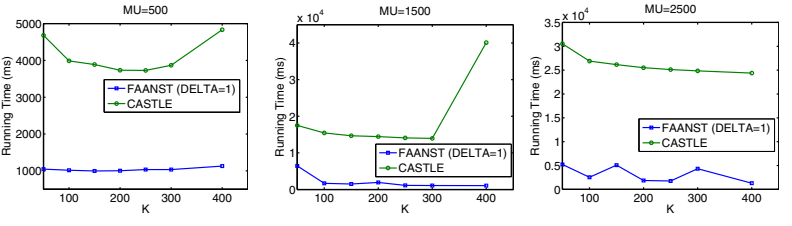


(c) Average number of suppressed tuples per *MU* = 500,1000,1500,2000, and 2500 vs. k on Dataset 2, Numeric data

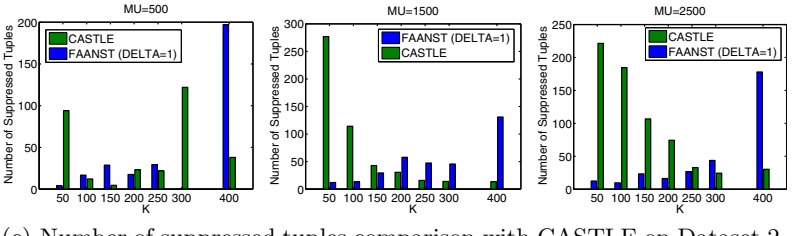
Fig. 3. Results of running FAANST on Adult Dataset



(a) Data loss comparison with CASTLE on Dataset 2, Numeric data



(b) Running time comparison with CASTLE on Dataset 2, Numeric data



(c) Number of suppressed tuples comparison with CASTLE on Dataset 2, Numeric data

Fig. 4. Results of comparison with CASTLE on Adult Dataset

5 Conclusions and Future Work

In this paper, we review some of the techniques for k-anonymity on both static and streaming data. We saw that most of the existing techniques for k-anonymity have been designed for static data, and are not suitable for streaming data requiring fast processing. So, we proposed FAANST which can anonymize numerical streaming data quite fast using window processing. The only problem which prevents FAANST from supporting categorical attributes is the clustering algorithm (k-means) we have used since this algorithm takes advantage of the concept of a centroid which is not definable on categorical values. To cope with this deficiency, we can use a medoid-based clustering algorithm such as one recently proposed by [13]. The concept medoid is easily definable on categorical attributes given a taxonomy tree for each attribute. According to the experiments we have done on both real and synthetic data, FAANST outperforms the existing techniques for anonymizing numerical streaming data. As future work, we are going to put a soft deadline on the delay each tuple can tolerate. Thus, tuples which are going to pass the soft deadline or those having already passed the soft deadline will be output with a higher priority, and will not remain in the system for a long time.

Acknowledgments

We express our gratitude to the authors of [3] for kindly providing us with their source code. The research of H.Z. was supported by a grant from the Natural Sciences and Engineering Research Council of Canada.

References

1. Asuncion, A., Newman, D.J.: UCI machine learning repository (2007), http://www.ics.uci.edu/~mlearn/
2. Bayardo, R.J., Agrawal, R.: Data privacy through optimal k-anonymization. In: Proceedings of the 21st International Conference on Data Engineering, USA, pp. 217–228 (2005)
3. Cao, J., Carminati, B., Ferrari, E., Tan, K.L.: Castle: A delay-constrained scheme for ks-anonymizing data streams. In: Proc. of the 2008 IEEE 24th ICDE, pp. 1376–1378 (2008)
4. Fung, B.C.M., Wang, K., Yu, P.S.: Top-down specialization for information and privacy preservation. In: Proceedings of the 21st ICDE, USA, pp. 205–216 (2005)
5. Hundepool, A., Willenborg, L.: mu and tau-argus: Software for statistical disclosure control. In: Proceedings of Third International Seminar on Statistical Confidentiality (1996)
6. Iyengar, V.S.: Transforming data to satisfy privacy constraints. In: Proc. of the Eighth ACM SIGKDD, pp. 279–288 (2002)
7. LeFevre, K., DeWitt, D.J., Ramakrishnan, R.: Incognito: efficient full-domain k-anonymity. In: Proc. of ACM SIGMOD, pp. 49–60 (2005)
8. LeFevre, K., DeWitt, D.J., Ramakrishnan, R.: Mondrian multidimensional k-anonymity. In: Proc. of the 22nd ICDE, p. 25 (2006)

9. Li, J., Ooi, B.C., Wang, W.: Anonymizing streaming data for privacy protection. In: Proceedings of the 2008 IEEE 24th International Conference on Data Engineering, USA, pp. 1367–1369 (2008)
10. Li, J., Wong, R.C.w., Fu, A.W.c., Pei, J.: Achieving k-anonymity by clustering in attribute hierarchical structures. In: Tjoa, A.M., Trujillo, J. (eds.) DaWaK 2006. LNCS, vol. 4081, pp. 405–416. Springer, Heidelberg (2006)
11. MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: Le Cam, L.M., Neyman, J. (eds.) Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297. University of California Press, Berkeley (1967)
12. Meyerson, A., Williams, R.: On the complexity of optimal k-anonymity. In: Proceedings of the Twenty-Third ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, pp. 223–228. ACM, New York (2004)
13. Park, H.-S., Jun, C.-H.: A simple and fast algorithm for k-medoids clustering. Expert Syst. Appl. 36(2), 3336–3341 (2009)
14. Patroumpas, K., Sellis, T.K.: Window specification over data streams. In: Grust, T., Höpfner, H., Illarramendi, A., Jablonski, S., Fischer, F., Müller, S., Patranjan, P.-L., Sattler, K.-U., Spiliopoulou, M., Wijsen, J. (eds.) EDBT 2006. LNCS, vol. 4254, pp. 445–464. Springer, Heidelberg (2006)
15. Samarati, P.: Protecting respondents' identities in microdata release. IEEE Trans. on Knowl. and Data Eng. 13(6), 1010–1027 (2001)
16. Samarati, P., Sweeney, L.: Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Technical report (1998)
17. Sweeney, L.: Datafly: A system for providing anonymity in medical data. In: Proc. of the IFIP TC11 WG11.3 Eleventh International Conference on Database Securty XI, pp. 356–381. Chapman & Hall, Ltd., Boca Raton (1998)
18. Sweeney, L.: k-anonymity: a model for protecting privacy. Int. J. Uncertain. Fuzziness Knowl.-Based Syst. 10(5), 557–570 (2002)
19. Wang, W., Li, J., Ai, C., Li, Y.: Privacy protection on sliding window of data streams. In: Proceedings of the 2007 International Conference on Collaborative Computing: Networking, Applications and Worksharing, Washington, DC, USA, pp. 213–221 (2007)
20. Xiao, X., Tao, Y.: Personalized privacy preservation. In: Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data, pp. 229–240. ACM, New York (2006)
21. Zakerzadeh, H.: Multi-degree anonymity in streaming data. Master's thesis, The University of Western Ontario (February 2010)
22. Zhou, B., Han, Y., Pei, J., Jiang, B., Tao, Y., Jia, Y.: Continuous privacy preserving publishing of data streams. In: EDBT, pp. 648–659 (2009)

Secret-Sharing Hardware Improves the Privacy of Network Monitoring

Johannes Wolkerstorfer

Telecommunications Research Center Vienna (FTW),
Donau-City-Strasse 1, 1220 Vienna, Austria
Johannes.Wolkerstorfer@ftw.at

Abstract. Network service providers monitor the data flow to detect anomalies and malicious behavior in their networks. Network monitoring inspects the data flow over time and thus has to store packet data. Storing of data impedes the privacy of users. A radically new approach counteracts such privacy concerns by exploiting threshold cryptography. It encrypts all monitored traffic. The used symmetric keys are made available to monitoring entities only if they collect enough evidence of malicious behavior. This new approach overcomes weaknesses of packet anonymization. It calls for dedicated hardware that is able to encrypt packets and generate key-share information for gigabit networks. This article proves that the application of Shamir's secret sharing scheme is possible. The presented hardware is able to protect up to 1.8 million packets per second. The creation of such a high-speed hardware required innovations on the algorithmic, the protocol, and on the architectural level. The outcome is a surprisingly small circuit that fits commercially available FPGA cards. It was tested under real-world conditions. It proved to protect the users' privacy while monitoring gigabit networks.

Keywords: Secret Sharing, Threshold Cryptography, Hardware Acceleration, Field-Programmable Gate Array (FPGA), Gigabit Ethernet.

1 Introduction

Traffic monitoring is a necessity for Internet service providers to guarantee accessibility and quality of services. Traffic monitoring usually captures the complete network traffic of all users. It seeks in the traffic for malicious behavior. It attempts detecting illegal intrusion into computer systems, distribution of viruses, and other harmful activities. Network monitoring helps improving the information security of customers but it also causes privacy problems because of data storage. Most monitoring applications require the storage of network data to detect malicious behavior. Malicious behavior is mostly not restricted to a single packet but spreads over multiple packets. Storage of network packets also stores possibly private data of users. This violates for instance the European legislation [6]. Even storing only the packet headers and dropping the payload causes privacy problems. The EU data-protection commission considers the IP address as

J. Garcia-Alfaro et al. (Eds.): DPM 2010 and SETOP 2010, LNCS 6514, pp. 51–63, 2011.

private information because it can identify individual users [6]. Obeying privacy legislation makes network monitoring a difficult task and calls for organizational and technical solutions.

The privacy aspects of network monitoring and possible technical solutions are addressed by recent research. The FP7 project PRISM investigated technical approaches for 'PRIvacy-aware Secure Monitoring' [2]. It proposed to heavily use symmetric encryption and threshold cryptography to solve the privacy problem. This article investigates the feasibility of the proposed approach and analyzes hardware aspects of the cryptographic hardware.

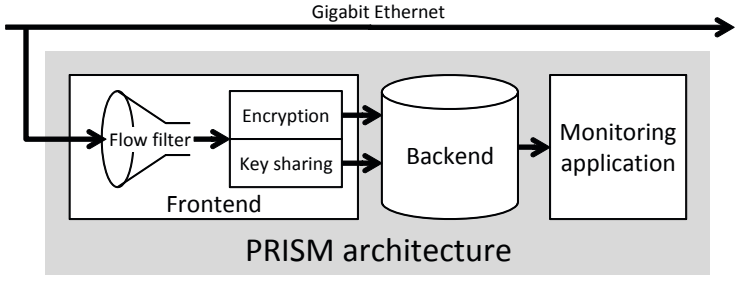


Fig. 1. The PRISM approach for privacy-preserving network monitoring

In contrast to most other attempts to solve the problem, the PRISM approach preserves the privacy not by anonymization but encrypts all the captured and filtered network traffic. R. Pang et al. give a good overview over the problems concerning packet anonymization [13]. The article of Burkhart et. al. shows that the possibility of injecting network traffic increases the chance of successful de-anonymization [5]. They come to the conclusion that anonymization techniques are not sufficient to protect data privacy. Figure 1 visualizes the new approach. It distributes the problem among three entities. The first entity is the *frontend*. It filters the traffic of interest and does the cryptographic operations. It sends the protected data to the *backend*. The backend basically stores the data and does not modify it. Monitoring applications, which are the third entity, can access the encrypted data. If they collect enough evidence that malicious behavior was going on they are able to re-construct the cryptographic keys which protect the captured packets.

This article focuses on the algorithmic, the architectural, and the implementation aspects of the cryptographic functions applied by the frontend. The requirements of the frontend are of particular interest because it receives data streams at gigabit rates. Though it is feasible to encrypt gigabit data streams with commodity hardware [17], this article shows for the first time that it is also possible to derive individual encryption keys for every of the hundred thousands of packets passing through every second. Moreover, the article shows that Shamir's secret sharing scheme is an adequate technique to raise the privacy level of network monitoring. Shamir's scheme allows the monitoring application not by policy

but by strict technical restrictions only to inspect those packets in plaintext for which enough key shares and thus evidence of malicious behavior have been collected. The article presents the first hardware implementation of Shamir's scheme with application in network security. It is a well optimized implementation that applies most efficient algorithms. It has a customized architecture that delivers astonishing throughput. The convincing performance numbers are not only shown on analytic level or by simulation. A fully-fledged implementation on the NetFPGA card is presented. The FPGA-based hardware can handle any shape of gigabit traffic. No software interaction except for configuration of master secrets and system parameters is needed.

The article recalls in Section 2 Shamir's secret sharing scheme, which protects secrets with key shares computed by evaluation of polynomials. In the same section (Sec. 2.1), we show how the Lagrange interpolation technique to re-compute secrets from key shares can be improved. The improvement saves 50% of the computation time. Section 3 presents the hardware architecture of the frontend. It concentrates on the novel Shamir secret sharing hardware. The results achieved in terms of throughput and hardware resources are presented in Section 4.

2 Shamir's Secret Sharing

Threshold cryptography is a cryptographic scheme where a trusted party generates $1 \leq m \leq n$ secret shares S_i of a secret S and distributes them [12]. When m (m is the threshold level) or more shares S_i are combined, the secret S can be revealed. No information about the secret S can be learned from collecting $m-1$ or fewer shares. Shamir's secret sharing scheme is a threshold scheme that fulfills these requirements [14]. It exploits the fact that a polynomial of degree $m-1$ is uniquely defined by m points on the polynomial. (1) defines the polynomial.

$$(x_i, y_i) = (x_i, P_f(x_i)) = \sum_{j=0}^{m-1} c_j x_i^j = c_{m-1}x_i^{m-1} + c_{m-2}x_i^{m-2} + \ldots + c_1 x_i + c_0 \quad (1)$$

A secret sharing scheme consists of two phases. First, key shares are computed. In a second step, any party possessing at least m different key shares can recompute the secret. In the case of Shamir's secret sharing scheme, the computation of key shares (x_i, y_i) is done by evaluating the polynomial $y_i = P_f(x_i)$. At least m pair-wise different points (x_i, y_i) have to be computed that the secret can be recovered later. In Shamir's secret sharing scheme, the secret is the least significant coefficient c_0 of the polynomial $P_f(x)$. The other coefficients c_i have to be chosen randomly. They have to be kept secret like c_0. Shamir's secret sharing scheme is indeed a threshold scheme because the knowledge of $m-1$ points on a polynomial of degree $m-1$ gives no information about the y-coordinate in $y = P_f(x = 0)$.

$$P_f(0) = \sum_{j=0}^{m-1} y_i l_i(0) = \sum_{j=0}^{m-1} y_i \prod_{j=0, j\neq i}^{m-1} \frac{x_j}{x_j - x_i} \tag{2}$$

An entity collecting m key shares can re-compute the secret. The original polynomial can be reconstructed from the m points (x_i, y_i) by polynomial interpolation. Lagrange interpolation shown in Equation (2) is an efficient approach [12]. Equation (2) shows a simplified version of Lagrange interpolation because only the last coefficient c_0 is of interest. c_0 is found at $x = 0$: $c_0 = P_f(0)$.

Table 1. Comparison of different interpolation algorithms

algorithm	time complexity	mults. m = 8	storage compl.
Newton	$2m^2 - 4m$ multiplications	96 (76,80%)	$\mathcal{O}(n^2)$
Neville-Aitken	$3m^2 - 3$ multiplications	183 (146,40%)	$\mathcal{O}(n)$
Lagrange original	$2m^2 - 3$ multiplications	125 (100,00%)	$\mathcal{O}(n)$
Lagrange improved	$m^2 + 2m - 3$ multiplications	77 (61,60%)	$\mathcal{O}(n)$

We evaluated several interpolation algorithms regarding computational and storage efficiency. The results are shown in the first three rows of Table 1. The Newton algorithm, which has best timing characteristics, is not suitable due to scalability problems of storage. The recursive Neville-Aitken approach was excluded because of long run time. So we analyzed the Lagrange algorithm for possible improvements.

2.1 Improved Lagrange Interpolation

Shamir's original secret-sharing scheme uses prime-field arithmetic. Using binary extension fields (GF(2^m)) improves the computational efficiency. GF(2^m) arithmetic has advantages for hardware implementation. This will be discussed later in Section 3. In GF(2^m), subtraction is identical to addition and thus turns the term $x_j - x_i$ in (2) into $x_j + x_i$. This slight change brings the possibility that many product terms in Equation (2) can be shared. Our in-depth analysis of the necessary calculation steps led to Equation (3). Although the time complexity is still quadratic, the quadratic term is halved. Time complexity was assessed in number of modular multiplications and neglected additions. All algorithms listed in Table 1 need one modular inversion.

$$P_f(0) = \left(\prod_{i=0}^{m-1} x_i \right) \left(\sum_{i=0}^{m-1} \frac{y_i}{L_i} \right) \quad \text{with} \quad L_i = x_i \prod_{j=0, j\neq i}^{m-1} (x_i + x_j) \tag{3}$$

The improved Lagrange interpolation Formula (3) looks more complicated at first sight but improves the efficiency a lot. It allows recovering a secret in 50% less time for high threshold levels m. Table 1 shows a comparison of the timing behavior for a threshold level of $m = 8$. Here, the performance gain of the new algorithm is 38.4%. During computation, the m input points (x_i, y_i), the m intermediate results L_i, and three variables have to be stored.

3 Customized Hardware Architecture

The architecture of the privacy-preserving network-monitoring approach, which was proposed by the PRISM project [2], is shown in Figure 1. The PRISM architecture is composed of three entities having a dedicated functionality. Basic idea of the PRISM architecture is to maintain the privacy of captured network traffic by disclosing information only to those observers (the monitoring applications in Fig. 1), who have collected enough evidence that malicious network traffic occurred. For the sake of brevity, we do not detail all aspects of the concept but concentrate on the cryptographic operations that protect the privacy of user data [2]. The task of the frontend is detecting and protecting network packets that are stored for later analysis. The flow filter selects packets from the data stream, which exhibit certain properties. A typical example of a flow filter is to pass all TCP traffic coming from a certain subnet. All other packets are dropped. In general, the flow filter decides stateless on a per-packet basis, which packets belong to a flow and which not.

Packets belonging to a flow are encrypted. They are stored in the backend for later inspection. The applied encryption mechanism is nearly identical to IPsec [10]. In this article, we do not go into the details of encrypting network traffic with AES-128 in CBC mode of operation. Please refer to [7,10,11,17]. Instead, we concentrate on the protection of the symmetric encryption key. In contrast to IPsec, the PRISM frontend does not negotiate encryption keys with the opposite party. This has two reasons: First, the monitoring application works usually offline. Second, the encryption key should be secret for the monitoring application until it has collected enough key shares.

At this point, the key sharing mechanism comes into play: If an anomaly of the filtered packets is detected, the frontend sends in addition to the encrypted packet key-share information to the backend. Anomalies are for instance patterns of distributed denial-of-service detected in packets. Often counting Bloom filters are used to identify anomalies [4]. The application of Snort rules in another method to detect anomalies [15]. The purpose of the backend is merely to store the encrypted packets and to regulate access for various monitoring applications. If a monitoring application can access more than m key-share information it is able to reconstruct the encryption key and thus to inspect the whole flow in plaintext. The configuration of the frontend (e.g. the choice threshold level m) has to assure a good tradeoff between privacy protection and monitoring precision.

As indicated by Figure 2, the computation of the key share in the frontend can happen concurrently to the encryption of the payload. The output of the key-share computation is appended at the end of encrypted packets. This lowers the overhead and allows placing the backend on another physical machine. A positive side-effect of communication solely based on Ethernet communication is that during development, entities can be first realized in software. They are replaced later by hardware without having the need to change interfaces.

The *keyshare* unit, which is of special interest in this article, is shown in Figure 2. Its challenges can be summarized as follows: Whenever an anomalous

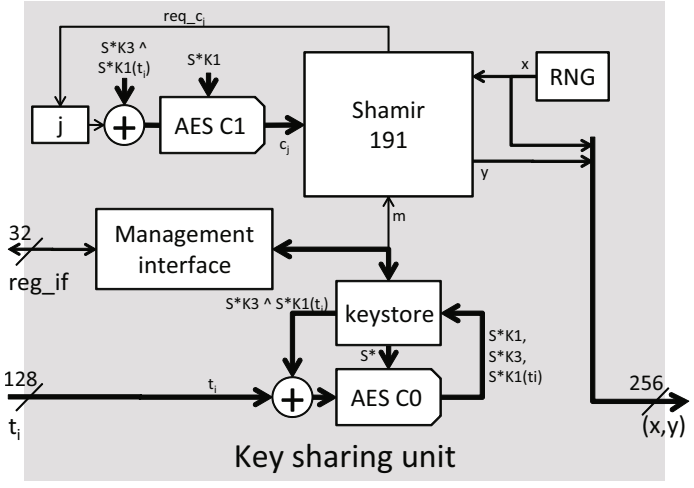


Fig. 2. Hardware architecture of the keyshare unit

flow packet gets encrypted the unit has to compute a key share by evaluating Equation (1). The key share protects the secret key which was used for packet encryption using the AES-128 algorithm. The data to protect is thus 128 bits. Hence, the Shamir polynomial must be defined over a finite field of order higher than 128 to assure exact re-construction of the original key after collecting m key shares. We have chosen the well-known binary extension field GF(2^{191}) mostly because of previous hardware experience with this field. It is used in elliptic-curve cryptography [1]. The finite field GF(2^{191}) is defined by the trinomial $f_{191}(x) = x^{191} + x^9 + 1$. Trinomials allow very efficient modular reduction in hardware. Many hardware implementations favor binary extension fields over prime fields.

$$P_f(x) = \sum_{j=0}^{m-1} c_j x^j = \Big(\big((0 + c_{m-1})x + c_{m-2}\big)x + \ldots + c_1\Big)x + c_0 \qquad (4)$$

In a straight-forward approach, polynomial evaluation would require $m-1$ squarings and multiplications when the coefficients are iteratively processed from c_0 to c_{m-1}. Processing the coefficients in the opposite direction from c_{m-1} to c_0 is more efficient. The so-called Horner scheme shown im Equation (4) is confident with $m-1$ multiplications. It has obviously linear time complexity. The hardware implementation of the Horner scheme computes in every iteration j: $y_j = y_{j+1} \cdot x + c_j$ with $j = m-1 \ldots 0$ and $y_m = 0$. Central operation in every iteration is the modular multiplication $y_{j+1} \cdot x$ in GF(2^{191}), which multiplies the intermediate result y_{j+1} by the chosen x-coordinate. The x-coordinate can be any element of GF(2^{191}). Its actual choice does not influence security as long as $x \neq 0$ because $P_f(0) = c_0$ would disclose the secret. The free choice of the x-coordinate allows an optimization, which was not reported in literature

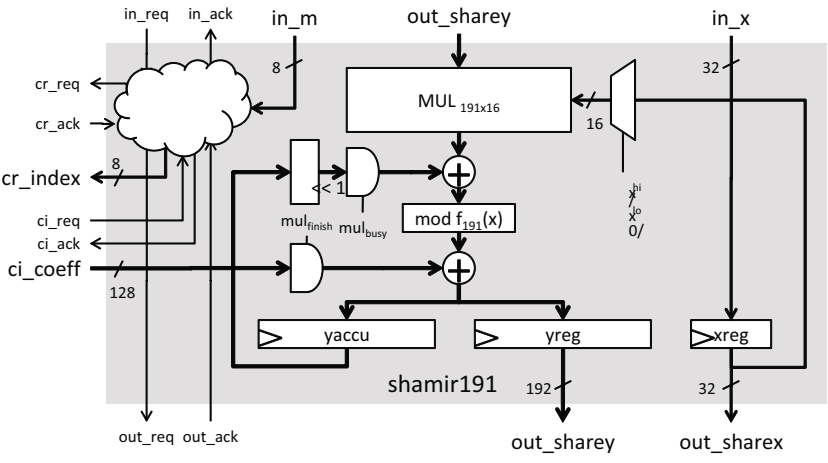


Fig. 3. Hardware architecture of the Shamir unit

before. The 191×191-bit multiplication $y \cdot x \mod f_{191}$ can be simplified when the precision of x can be reduced. The precision of x in terms of word width must suite the maximum threshold level m: $x \geq m$. So at least m different key shares can be computed. It is expected that at most 1024 key shares will make sense in practice (m: 10 bits). Thus, restricting the x-coordinate to 32 bits gives a generous margin. 32-bit x-coordinates reduce the original 191×191-bit multiplication to a 191×32-bit multiplication. This lowers the hardware resources by a factor of six or improves performance by 500%. We decided for a trade-off between both options.

The finite-field multiplier is the central component of the *shamir191* hardware architecture shown in Figure 3. It applies digit-parallel multiplication to implement the 191×32-bit multiplication in $\mathrm{GF}(2^{191})$. The intermediate value y_{j+1} (*yreg*) is scheduled at full-precision (parallel), while the x-coordinate is split into two 16-bit digits: $x = x_{hi}2^{16} + x_{lo}$. The required 191×16-bit multiplier is a very good trade-off between hardware resources and performance. It needs two clock cycles for computing $y \cdot x \mod f_{191}$. A 191-bit x-coordinate would require 12 cycles. The multiplier applies interleaved modular reduction. It immediately reduces every 206-bit intermediate polynomial product. The modular reduction circuit is a small circuit requiring a couple of XOR gates. This efficient implementation is only possible because a fixed finite field is used. The choice of a trinomial as irreducible polynomial minimizes the number of XOR gates needed.

The *shamir191* hardware shown in Figure 3 evaluates a Shamir polynomial in an iterative manner. At the start, the x-coordinate and the threshold level m are loaded. The finite-state machine of *shamir191* starts immediately with the 2-cycle modular multiplication and requests the highest coefficient c_{m-1}. Coefficients c_j are provided by the external AES unit C1 as shown in Figure 2. The *shamir191* unit requests them via the coefficient-request *cr_* interface. When the computation of a coefficient c_j has finished it is delivered via the *ci_* interface.

This rather complicated interface has two reasons: First, it clearly separates the evaluation of the Shamir polynomial from coefficient generation. Second, the interface does not make any assumptions on the latency and throughput capabilities of the coefficient generation. When the coefficient computation is able to deliver results within two clock cycles, the *shamir191* unit has maximum performance. When it takes longer, *shamir191* will insert wait states. The *cr_* and the *ci_* interfaces are fully decoupled. This allows computing coefficients in a pipelined manner. Such a pipelined architecture with low latency requirements but high throughput is useful in practice and requires no modifications of *shamir191*'s coefficient interface. The current hardware uses only a single pipeline stage (AES C1) because this approach already fulfills the performance requirements as Section 4 will show.

The computation of the coefficients of the Shamir polynomials is the second big challenge of the key-share hardware besides fast evaluation of $P_f(x_i)$. The coefficients have to be computed sequentially from index $j = m - 1$ down to $j = 0$. The last coefficient c_0 is the secret to protect. It is used for the symmetric encryption of flow packets. As stated before, the Shamir polynomial and thus its coefficients have to be unique for every packet. Packets are identified by a 128-bit value t. t is called the flow identifier. Storing a set of coefficients for each flow identifier is not feasible. Thus, coefficients have to be computed on a per-packet basis. The PRISM architecture makes use of keyed hash functions to derive coefficients from the flow identifier t. The security of the algorithm relies on the secrecy of the master secret S^*. S^* is rather static. The basic idea is to use the keyed hash function to derive the key $c_0 = H(S^*, t)$. This assures the secrecy of S^* even when c_0 gets disclosed. The computation of the key c_0 is generalized to the computation of all coefficients of the Shamir polynomial: $c_j = H(S^*, t|j)$. The term $t|j$ denotes concatenation of the 128-bit flow identifier t and an index j, which is a 16-bit value. For sake of lower hardware complexity and better design reuse, the AES-XCBC-PRF-128 function was selected to compute the coefficients. AES-XCBC-PRF-128 is used in the Internet key-exchange protocol of the IPsec standard [9]. The application of this function is shown in Equation (5). AES-XCBC-PRF does not use the secret S^* directly but derives the keys S^*_{K1} and S^*_{K3} by encryption of the constant 128-bit values 0x01010101010101010101010101010101 and 0x0303 ... 03. The computation of S^*_{K1} and S^*_{K3} is only required, when the master secret S^* changes.

$$\begin{aligned} c_j = \mathrm{AESXCBC}(S^*, t|j) = \mathrm{AES}(S^*_{K1}, \mathrm{AES}(S^*_{K1}, t) \otimes S^*_{K3} \otimes \mathrm{PAD}(j)), \\ S^*_{K1} = \mathrm{AES}(S^*, 0x010101..01), \quad S^*_{K3} = \mathrm{AES}(S^*, 0x030303..03) \end{aligned} \tag{5}$$

The coefficient computation, which has to be done on a per packet basis comprehends $m + 4$ AES encryptions. First, the flow identifier t has to be encrypted by the encryption key S^*_{K1}. The resulting ciphertext is chained with S^*_{K3}. The term $S^*_t = \mathrm{AES}(S^*_{K1}, t) \otimes S^*_{K3}$ is static for the computation of all coefficients c_j. Four AES encryptions are necessary as pre-computation to start the

computation of the coefficients c_j. After this, m AES encryptions can compute the m coefficients: $c_j = AES(S^*_{K1}, S^*_t \otimes \mathrm{PAD}(j))$. AES-XCBC-PRF makes use of 10* ('ten star') padding, which inserts a '1'-bit after the left-aligned 16 bits of j.

The computational effort to calculate coefficients grows linear with the number of coefficients m. It has linear timing complexity as the evaluation of the Shamir polynomial has. It can be parallelized because the plaintext $S^*_{K1}, S^*_t \otimes \mathrm{PAD}(j)$ does not depend on previous coefficients. Another advantage that can be utilized by a hardware implementation is the fact that $m + 1$ encryptions use the same round keys derived from S^*_{K1}. Multiple AES units can share this information. Our hardware (Figure 2) uses only one round-key generator that is shared between the AES cores C0 and C1.

4 Implementation Results

The Shamir secret sharing scheme was realized on the NetFPGA card [16]. The NetFPGA is a network interface card, which features four Gigabit Ethernet interfaces that are accessible under Linux. Central element of the NetFPGA card is a Virtex-II Pro 50 FPGA [18], which can be configured to process the gigabit data streams. The NetFPGA project is an open-source project, which delivers source code for a network interface card, a network switch, and a router out of the box. The broad support of NetFPGA by a global developer community simplifies the setup of projects and allows concentrating on specific problems—novel Shamir key-sharing hardware for gigabit rates in our case.

The implementation of the Shamir secret sharing scheme requires a strict methodology to create working hardware that achieves highest throughput using least resources. The hardware was realized in three steps. After fixing the functional requirements, an executable specification was implemented. This C/C++ implementation mimics the functionality of hardware. It reads Ethernet packets, analyzes its contents and does the required cryptographic operations. The C/C++ model helped to remove any ambiguities from the specification and showed the feasibility of the concept.

In the second step of realization, the key sharing unit was described with the hardware-description language Verilog. The Verilog model differs widely from the functional C/C++ model because it is more a structural and hierarchical representation of the hardware. This model implements the architecture shown in Figure 3 and in Fig. 2. It is very efficient in terms of hardware resources needed. The resource efficiency is due to the consequent reuse of hardware modules. For instance, the AES unit *C0* is used to derive encryption keys and does the bulk encryption of packet payloads.

Simulation assured correct functionality of the Verilog model. The whole model of the *keyshare* unit was simulated on the *user_data_path* packet level. The *user_data_path* is an abstraction layer of the NetFPGA project, where Gigabit Ethernet data streams are represented as 64-bit streams in a single 125 MHz clock domain. This simulation model is close to the physical layer and hence

Table 2. Results of the hardware implementation on a Virtex-II Pro 50 FPGA

module	**slices**	**BRAM**	**clock**	$\mathbf{f_{max}}$	**performance**	**throughput**
name	instances	instances	cycles	[MHz]	max [ops/s]	[Mbps]
shamir191	1633 (7%)	0 (0%)	$2m+3$	210.6	$12.3 \cdot 10^6$	2359
AES-128	417 (2%)	10 (4%)	11	289.1	$26.3 \cdot 10^6$	3364
Keyshare unit	3687 (16%)	18 (7%)	$11m+3$	163.5	$1.8 \cdot 10^6$	343

accurate. Nevertheless, it allows rather simple input of test packets, which were generated by the C/C++ model. Besides functional correctness, the Verilog simulation also verified the desired timing in terms of clock cycles.

The Verilog model was also used for synthesis, which maps the hardware functionality onto resources of the Virtex-II Pro 50 FPGA [18]. This device is a medium-sized field-programmable gate array that is the central element of the NetFPGA card [16]. The external DDR2 memory resources of the NetFPGA card were not used for this project. Just access to the physical layer (a Broadcom chip [3]) and access to the PCI bus are utilized. The PCI bus makes the management interface (see Figure 2) accessible under Linux. Via this interface, cryptographic keys and other parameters (like the threshold level m) can be configured.

Table 2 summarizes the results of the hardware implementation on the Virtex-II Pro device. In total, only 16% of the device are utilized. This resource efficiency is due to the rigorous reuse of hardware components. Reusing resources is more efficient but requires more attention during development because of complicated dataflow and complex finite-state machines, which control the streams. The synthesis results in Table 2 also show the clever utilization of the FPGA resources. The Shamir unit requires only FPGA slices for realizing the big combinational function. The unit needs no block RAM (BRAM) resources. Contrary, the two instances of AES encryption engines (*AES C0* and *AES C1*) predominantly use these 18-kbit BRAM blocks. Eight BRAM blocks can implement 16 synchronous AES T-Tables by exploiting the dual-port facility. As shown in [17,11], the rest of the AES encryption hardware is mostly XORing T-Table outputs and key addition, which is also a XOR operation. The exploitation of two different and thus orthogonal resources (slices, BRAM) influences the hardware efficiency positively. Table 2 indicates the resource utilization of the Virtex-II Pro 50 device as percentage in parentheses. In total, the cryptographic unit consumes only 16% of the available resources. This means that the FPGA could host a crypto unit for each of the four physical gigabit interfaces.

The hardware implementation of the Shamir secret sharing scheme has astonishing performance. The performance and throughput values given in Table 2 show the capabilities when clocked at the maximum clock frequency f_{max}. The *shamir191* unit, which computes the key shares by evaluation of the Shamir polynomial, can compute 12.3 million key shares (x_i, y_i) per second ($m = 8$). This performance assumes that the coefficients of the polynomial can be computed within two clock cycles. The current hardware architectures supports only one AES-128 unit, which takes 11 clock cycles for that operation. Thus, the computation time for a key share extends from $2m + 3$ to $11m + 3$ clock cycles. When

choosing a threshold level of $m = 8$, a key-share computation takes 91 clock cycles. The top-level circuit *keyshare* unit, which combines *shamir191*, two instances of *AES-128*, and control machines can be clocked at 163 MHz. It allows to process 1.8 million key shares per second. At the target clock rate of 125 MHz, 1.3 million key-share computations are possible. This outstanding performance allows computing an individual key share for every packet coming over Gigabit Ethernet. Even when the whole available bandwidth is consumed by very small packets (64 byte), the *keyshare* unit can cope with this situation.

Real network measurements confirmed the excellent performance predicted by simulation and synthesis. The implementation of a working prototype on the NetFPGA card occupies 32% of its resources (32% of the slices, 7% of the BRAMs). The additional resources in comparison to Table 2 are caused by components of the NetFPGA interfaces (media-access controller, PCI interface, FIFOs). Our test setup consists of a Linux PC that hosts the NetFPGA card. An additional network interface (eth1) generates network traffic that is processed by the NetFPGA. The output of the card was captured by a second PC with Wireshark to disburden the Linux machine. In the field test, the same testdata as in the Verilog simulation was used. This assures functional correctness. The Linux tool *tcpreplay* was used for sending two different testdata sets. One set contains 1000 very short packets (roughly 100 bytes). The other set contains 1000 large packets (roughly 1000 bytes each). tcpreplay achieved sending 50,000 packets per second causing a throughput of 38.15 and 381.47 Mbps, respectively. All tests indicated correct functionality. The Verilog simulation assures correctness also at gigabit line rates, but this situation with throughput rates approximately around 800 Mbps, was not producible with PC-based equipment.

5 Conclusions

This article proved that threshold cryptography is a viable technique to improve the privacy aspects of network monitoring. The article investigated the question whether Shamir's secret sharing scheme is applicable in practice. The actual question of interest was: Can the Shamir's scheme be applied at an FPGA-based frontend that captures gigabit traffic? This article can answer this question not only with clear 'yes' but also presents a working prototype that can protect any shape of Gigabit Ethernet traffic. It is the first documented hardware implementation of Shamir's scheme. Bringing forth this prototype, which consumes only 16% of the Virtex-II Pro 50 FPGA on the NetFPGA card, required scientific advances on the algorithmic, on the protocol, and on the architectural level.

The algorithmic improvements concern the optimization of Lagrange interpolation algorithm over binary extension fields which saves 50% of the multiplications for recovering keys in monitoring applications. An improvement on the protocol level is the use AES-XCBC-PRF function as keyed hash function to derive session keys for every packet. The coefficients of the Shamir polynomial are computed in the same way. A couple of AES-XCBC-PRF computations are necessary per packet. Insofar, it is most important that this function can be

computed efficiently in hardware. A novel pipelined coefficient-request interface of the Shamir unit allows computing the function in parallel or in pipelined fashion. The Shamir unit itself picks up ideas which make hardware implementations of elliptic-curve cryptography faster. It uses a digit-parallel multiplier with interleaved modular reduction for operation in GF(2^{191}). It can reach a maximum throughput of 12.3 million key-share computations per second. This number exceeds the maximum number of packets on Gigabit Ethernet by a factor of more than ten. The whole cryptographic hardware is very small. The 3687 slices and the 18 BRAM blocks occupy only 16% of the resources of the Virtex-II PRO 50 device. The circuit was tested in practice. A realization on the NetFPGA card proved correct functionality for different traffic scenarios of Gigabit Ethernet.

This first very successful hardware implementation of Shamir's secret sharing scheme will call for additional areas of application because it proved that very small hardware can share secrets at incredible rates. Our future activities will improve the results even further. We will utilize the pipelined coefficient-request interface to attach an AES-XCBC-PRF unit with four pipeline stages. This allows exploiting the full throughput capability of the hardware for polynomial evaluation. Another improvement of the presented approach will be the hardening of the secret-sharing mechanism against intentional or accidental modification of shared secrets. Such modification would lead to wrong recovery of protected keys but might be countered by rather simple measures like adding redundant shares [8].

Acknowledgements

We thank the researchers at the University Tor Vergata in Rome for the fruitful discussions and their grateful support.

References

1. American National Standards Institute (ANSI). AMERICAN NATIONAL STANDARD X9.62-2005. Public Key Cryptography for the Financial Services Industry, The Elliptic Curve Digital Signature Algorithm, ECDSA (2005)
2. Bianchi, G., Teofili, S., Pomposini, M.: New Directions in Privacy-Preserving Anomaly Detection for Network Traffic. In: Antonatos, S., Bezzi, M., Boschi, E., Trammell, B., Yurcik, W. (eds.) NDA, pp. 11–18. ACM, New York (2008)
3. Broadcom. BCM5464SR Quad-Port Gigabit Copper Transceiver with Copper/ Fiber Media Interface (2006), `http://www.broadcom.com/products/Physical-Layer/Gigabit-Ethernet-PHYs/BCM5464SR`
4. Broder, A.Z., Mitzenmacher, M.: Network Applications of Bloom Filters: A Survey. Internet Mathematics 1(4) (2003)
5. Burkhart, M., Schatzmann, D., Trammell, B., Boschi, E., Plattner, B.: The Role of Network Trace Anonymization Under Attack. SIGCOMM Comput. Commun. Rev. 40(1), 5–11 (2010)
6. EU Article 29 Data Protection Working Party. Opinion on the Concept of Personal Data (01248/07/EN WP 136) (April 2007)

7. Frankel, S., Glenn, R., Kelly, S.: RFC 3602: The AES-CBC Cipher Algorithm and Its Use with IPsec. RFC 3602 (Proposed Standard) (September 2003)
8. Harn, L., Lin, C.: Detection and Identification of Cheaters in (t, n) Secret Sharing Scheme. Designs, Codes and Cryptography 52, 15–24 (2009), doi:10.1007/s10623-008-9265-8
9. Hoffman, P.: RFC 3664: The AES-XCBC-PRF-128 Algorithm for the Internet Key Exchange Protocol, IKE (2004)
10. Hoffman, P.: RFC 4308: Cryptographic Suites for IPsec. RFC 4308 (Proposed Standard) (December 2005)
11. Lemsitzer, S., Wolkerstorfer, J., Felber, N., Braendli, M.: Multi-gigabit GCM-AES Architecture Optimized for FPGAs. In: Paillier, P., Verbauwhede, I. (eds.) CHES 2007. LNCS, vol. 4727, pp. 227–238. Springer, Heidelberg (2007)
12. Menezes, A.J., van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography. Series on Discrete Mathematics and its Applications. CRC Press, Boca Raton (1997) ISBN 0-8493-8523-7, http://www.cacr.math.uwaterloo.ca/hac/
13. Pang, R., Allman, M., Paxson, V., Lee, J.: The Devil and Packet Trace Anonymization. SIGCOMM Comput. Commun. Rev. 36(1), 29–38 (2006)
14. Shamir, A.: How to Share a Secret. Communications of the ACM 22(11), 612–613 (1979)
15. Song, H., Sproull, T.S., Attig, M., Lockwood, J.W.: Snort Offloader: A Reconfigurable Hardware NIDS Filter. In: Rissa, T., Wilton, S.J.E., Leong, P.H.W. (eds.) FPL, pp. 493–498. IEEE, Los Alamitos (2005)
16. Stanford University. NetFPGA Project. NetFPGA (2009), http://netfpga.org/
17. Wolkerstorfer, J., Szekely, A., Lorünser, T.: IPsec Security Gateway for Gigabit Ethernet. In: Ostermann, T. (ed.) Austrochip 2008 – Proceedings of the 16th Austrian Workshop on Microelectronics (October 2008)
18. Xilinx Corporation. Virtex-II Pro and Virtex-II Pro X Platform FPGAs: Complete Data Sheet (2007), http://www.xilinx.com/support/documentation/virtex-ii_pro_data_sheets.htm

Non-uniform Stepping Approach to RFID Distance Bounding Problem*

Ali Özhan Gürel[1], Atakan Arslan[1,2], and Mete Akgün[1,3]

[1] TÜBİTAK UEKAE, 41470, Gebze, Kocaeli, Turkey
[2] Department of Electronics and Communications,
Faculty of Electrical and Electronics Engineering,
İstanbul Technical University, İstanbul, Turkey
[3] Computer Engineering Department,
Boğaziçi University, İstanbul, Turkey
{aligurel,atakana,makgun}@uekae.tubitak.gov.tr

Abstract. RFID systems are vulnerable to relay attacks (*mafia fraud* and *terrorist fraud*) as well as *distance fraud*. Several distance bounding protocols suitable to RFID systems were proposed to avoid these attacks. The common point of these protocols is to try to reduce success probability of the attacker. To the best of our knowledge, there is no RFID distance bounding protocol without final signature that provides success probability of attacker smaller than $(3/4)^n$ in the presence of all frauds. In this paper, we propose an RFID distance bounding protocol that creates binary responses by traversing the register with non-uniform steps based on the secret key in addition to binary challenges. Our protocol without final signature is the first to converge the success probability of the attacker to the ideal case, which is $(1/2)^n$ for all frauds. Furthermore, our protocol is robust against disturbances of channel, has low computational cost and also provides privacy.

1 Introduction

Radio Frequency IDentification (RFID) technology, including contactless smart cards are popularly used in many applications that need security concerns such as payment systems (public transport, ticketing) or passes (ID cards, company pass, passports) [8]. They do not have internal power source and the required energy is obtained by the transmitted signal of the reader. RFID systems are also low-cost devices which have restricted capacity and limited computational power.

The primary purpose of the security protocols in RFID systems are identification and authentication. Numerous security protocols using conventional cryptographic models have been proposed. However, these protocols are vulnerable to relay attacks, in which an adversary between a tag and a reader attempts

* This work has been partially founded by FP7-Project ICE under the grand agreement number 206546.

J. Garcia-Alfaro et al. (Eds.): DPM 2010 and SETOP 2010, LNCS 6514, pp. 64–78, 2011.

to authenticate herself by relaying messages between both parties. There are basically two sorts of relay attacks: *Mafia fraud* and *terrorist fraud*. Mafia fraud was first described by Desmedt et al. in 1987 [6]. In this attack scenario, an adversary receives the interrogation of a valid reader and passes it to a valid tag. It then receives the response from the tag and passes it to the reader so the adversary deceives the valid reader without awareness of the tag.

Terrorist fraud is similar to mafia fraud. In this attack scenario, the legitimate tag is dishonest and helps a illegal tag to convince the legitimate reader without giving its private key. Therefore, whenever the adversary wants to perform this attack, she always needs help of the valid tag.

The attacks seem practical to the above mentioned RFID applications. For instance, highway tolls are typically collected using RFID devices such as active tags that are on the vehicles. To pass the tolling point, the customer must be close enough to the reader ($\boldsymbol{R}$) of the toll point. A tag can be authenticated only if it is inside the interrogation zone of the reader. Assume that an attacker car ($\boldsymbol{A}$) is in front of a victim car ($\boldsymbol{V}$) in a queue. $\boldsymbol{A}$ successfully passes the toll point without paying by only relaying the messages between $\boldsymbol{R}$ and $\boldsymbol{V}$. This scenario is a practical example for mafia fraud. A successful mafia attack was first performed by Hancke [9]. Later, practical and low cost relay attacks on the contactless smart cards are defined and vulnerabilities of the cards are presented in the literature [9,12]. Theoretically developments have been experimented in [7], and have been adopted in real-life.

Similar to the relay attacks described above, there is also *distance fraud* attack in which the legitimate tag deceives the legitimate reader as if she pretend to be inside the legal zone (or neighborhood of the reader). In distance fraud, a lonely prover himself is assumed to be dishonest. Distance fraud is different from other frauds because there is no tag impersonation in this attack scenario.

Determining an upper bound on the distance between a reader (verifier) and a tag (prover) for authentication is a clever solution to defense relay and distance attacks. For this reason, several distance bounding techniques are developed within a defined particular range for a secure authentication. Measuring the received signal strength (RSS), the angle of arrival (AoA) and the round trip time (RTT) are the distance bounding methods to estimate an upper bound on the distance. The RSS and AoA's methods are not secure because an attacker can use a directional antenna and transmit more powerful signal to authenticate herself when she is even far from the legitimate distance. Therefore, various distance bounding protocols based on RTT measurement, are proposed to accomplish to this bounding problem [10,3,22,18,23]. The protocols allow measuring the time intervals between challenges and responses to upper-bound the distance between prover and verifier with respect to the speed of light.

In this paper, we propose a novel secure RFID distance bounding protocol which provides the best known security features among the existing ones. Our protocol is different in following sense. When creating binary responses, our protocol traverses the register with non-uniform steps based on the secret key and binary challenges. Our protocol is the first to converge to the success probability

of the attacker to the ideal case, which is $(1/2)^n$ for mafia, terrorist and distance frauds. Furthermore, it is resistant to channel errors and provides privacy and has good performance in terms of memory and computation requirements.

The rest of this paper is structured as follows: In Section 2, we briefly review some of the existing distance bounding protocols. In Section 3, we describe our protocol. Section 4 presents its security and performance analyses. In Section 5, we compare our protocol with the previous ones and at last we conclude the paper.

2 Related Work

In 1993, Brands and Chaum [3] propounded the first distance bounding protocol to prevent mafia fraud. Mafia fraud resistance is achieved by verification of the physical proximity of the prover by means of a rapid bit exchange phase. During this phase, the verifier sends a random challenge bit and the prover responds it immediately with a random response bit. This step is repeated n times where n is a security parameter. The round-trip time for all bit exchanges are measured and the signature sent by the prover that includes all these bits is checked by the verifier. Since a mafia attacker can not form this signature, her success probability is $(1/2)^n$. However, the prover can perform a distance fraud by sending the response bits before the verifier sends the challenge bits since they are independent. Brands and Chaum solved this problem by making the challenge bits and response bits dependent to each other by the help of a commitment phase. By means of this dependence, the prover can not predict the challenges and send the corresponding responses earlier. As a result, her success probability is also $(1/2)^n$.

In 2003, Capkun et al. [5] modified Brands and Chaum's protocol [3] to provide mutual authentication and called it as MAD. In their protocol, unlike in [3], both parties make commitments for their random bit strings and use these strings in rapid bit exchange. During the rapid bit phase, one party creates a random bit and XOR'es it with the previous bit coming from the other party. During each step, round-trip time is checked by both parties. Afterwards, both sides compute MAC of their identities concatenated with the bits that are used during rapid bit phase and send these MAC's to each other for verification. This protocol is not resilient to bit errors during the rapid bit phase and contains four computations at each side. Additionally, it does not prevent terrorist attack.

In 2005, Hancke and Kuhn (HK) [10] proposed a distance bounding protocol that has a different working principal than the protocol in [3]. In their first protocol, the verifier sends a nonce to the prover. Both parties then compute a MAC value of this nonce that will be used in rapid bit phase. During fast bit phase, the verifier sends random challenge bit and the prover replies with a response bit dependent on both the challenge bit and corresponding bit of MAC computed earlier. In this protocol, the success probability of distance fraud is $(3/4)^n$ whereas it is not resistant to mafia and terrorist fraud. In their previous protocol, mafia attacker can interrogate the prover and learns the contents of

MAC completely before rapid bit phase. This problem is solved by adding an extra step in which the prover sends a nonce to the verifier. Thus, like the distance fraud, the mafia attacker has a success probability of $(3/4)^n$. Thereafter, in [11], the authors present extensions for HK protocol to enhance the security against mafia and distance fraud without additional cost. They also show the trade-off between the security levels of mafia and distance frauds for HK-like protocols. However, they do not give any solution for terrorist fraud.

In [21], Singelée and Preneel explained two solutions to terrorist attacks. First of them is to use the secret key in the fast bit exchange which is the idea of Bussard in [4]. The second method is to use the trusted hardware that makes impossible for an attacker to extract out the values stored. However, this idea looks impractical for RFID systems.

In 2007, Reid et al. [20] combined the modified version of the Hancke and Kuhn's protocol [10] with the idea of Bussard [4] for terrorist resistance. Although the success probability of distance, mafia and terrorist fraud is $(3/4)^n$, which is optimum values for these attacks at that time, the privacy is not preserved as the ID of both parties are sent in clear form.

In [22], Singelée and Preneel designed a noise resilient mutual distance bounding protocol by modifying the Capkun's protocol [5] by the help of the noise-resilience property of the Hancke and Kuhn's protocol [10]. However, in 2008, Munilla and Peinado [17] carried out effective relay attacks on that protocol which increase the success probability of an adversary.

In [23], Tu and Piramithu proposed a distance bounding protocol in order to reach a minimum success probability for an attacker. They utilized the method used in Reid et al.'s protocol [20] against terrorist attack which was first mentioned in [4]. The rapid bit phase is divided into four sections. At the end of each section, the verifier must authenticate herself to the prover by means of sending a MAC value. As the number of sections increases, the number of verifications that are performed by the verifier increases. As a result, the success of the attacker decreases, but the computational cost of the protocol increases. Kim et al. [14] showed that this protocol is not secure against the key recovery attacks.

In 2008, Munilla and Peinado [16] modified the Hancke and Kuhn's protocol [10] in order to decrease the success probability of the mafia attack. It is accomplished by using void-challenges that the verifier deliberately leaves without sending. Since the mafia attacker does not know which of the challenges are void, she can not interrogate the prover successfully and learn even the half of the contents of the MAC value. According to their calculation, the optimum value for the probability of a challenge being full is $(4/5)$ and for this situation, the probability of an adversary's success is $(3/5)^n$. However, as stated in [13,14], three physical states (0, 1 and void) are difficult to implement. The authors also propose a modified version of this protocol to present better performances but in [1], it is shown a more precise upper bound than the one found in [16] for mafia fraud.

In [14], Kim et al. proposed a distance bounding protocol that provides privacy, channel error resistance and optionally mutual authentication. This protocol

includes similar messages and computations with Reid et al.'s protocol [20]. There are also differences in Kim et al.'s protocol with [20]: MAC value is created only with the nonce generated by the prover, ID of both parties are not sent publicly and a signature is sent after rapid bit phase. At the end of the protocol, the verifier calculates the error values stemming from three sources: differences between challenges sent and challenges received, differences between responses desired and responses given and also late responses. If the total error is larger than the fault tolerance threshold, the verifier aborts the process. Optionally, in the final step, the verifier sends MAC value of the prover's nonce in order to authenticate herself to the prover. Considering mafia fraud, an adversary can produce a valid response with $(1/2)$ success probability to the corresponding challenge bit. On the other hand, distance fraud and terrorist fraud can succeed with the probability $(3/4)^n$. In [19], a passive full disclosure attack to this protocol is presented. In this attack, the attacker discovers the long-term secret key of the prover by listening the public messages exchanged on the channel.

In [18], Nikov and Vauclair designed a distance bounding protocol that is based on rapid message exchange. They used more than one bit in the rapid exchange phase and moreover their protocol is not resilient to channel noise.

In 2009, Avoine and Tchamkerten [2] propounded an efficient distance bounding RFID authentication protocol. In this protocol, both parties compute a $2^{n+1} + m - 2$ bit MAC from random nonces sent to each other where m and n are credential size for authentication and number of rounds for the proximity check, respectively. First m bits of this MAC value are used for authentication whereas the rest of the bits are for proximity check. The authors made a balancing relation between false-acceptance ratio and memory requirement by utilizing multiple trees. However, their protocol is not resilient to bit errors during rapid bit phase.

In [13], Kim and Avoine modified the Hancke and Kuhn's protocol [10] in order to decrease the probability of mafia fraud. In this protocol, both random challenges and predefined challenges are sent by the verifier. Random challenges are not predicted by the prover before the verifier sent them whereas the predefined ones are known to both parties. By means of these predefined challenges, a mafia attacker can not interrogate the prover to learn the contents of the MAC before the rapid bit phase starts. As a result, the success probability of mafia fraud is almost $(1/2)^n$. However, the distance fraud can succeed with probability of $(7/8)^n$. Also, their protocol is not resistant to terrorist attack.

3 Proposed Protocol

The protocol described in this section consists of a slow nonce-exchange phase, a rapid bit exchange phase and a slow verification phase. As shown in Figure 1, the verifier and the prover share an n-bit secret x and this secret is assumed not to be revealed by neither parties. For the sake of simplicity, we use the abbreviation V for the verifier and P for the prover.

Table 1. Notations

$\sharp a$	: number of a
$a \| b$	: concatenation of the messages a and b
n	: number of rapid-bit exchange
x	: n-bit secret shared between the verifier and the prover
N_a, N_b	: Random nonces generated by the verifier and the prover, respectively
f	: one-way and collusion resistant pseudo-random function
f_x	: one-way and collusion resistant pseudo-random keyed function
R	: $2n$-bit output of the pseudo-random keyed function with inputs N_a and N_b and key x
j_1	: cursor that points the left edge bit of the secret x at the beginning
j_2	: cursor that points the right edge bit of the secret x at the beginning
k_1	: cursor that points the left edge of the register R at the beginning
k_2	: cursor that points the right edge of the register R at the beginning
c_i	: i^{th} challenge sent by the verifier
c_i'	: i^{th} challenge received by the prover
r_i'	: i^{th} response sent by the prover
r_i''	: i^{th} response received by the verifier
Δt_i	: i^{th} time difference between the challenge sent and the response received by the verifier
Δt_{max}	: maximum amount of time for the challenge sent and response received by the verifier
T_t	: number of tolerances for Δt_i's that exceed Δt_{max}
T_c	: number of tolerances for difference between the challenge sent by the verifier c_i and received by the prover c_i'
T_r	: number of tolerances for difference between the response sent by the prover r_i' and received by the verifier r_i''

Our protocol is as follows:

1. V generates a random n-bit nonce N_a and sends it to P. Similarly, P generates a random n-bit nonce N_b and sends it to V.
2. Both parties compute $2n$-bit MAC, $f_x(N_a, N_b)$, from these nonces by using the secret x as the key of the one-way and collusion resistant pseudo-random function, f, and assign these values to the register R. Recent pseudo random number generator (PRNG) model for RFID devices is presented in the literature [15].
3. As soon as the computation of the register is completed, P sets the cursor k_1 to the most significant bit (i.e., left edge) of the register R, and the cursor k_2 to the least significant bit (i.e., right edge) of the register R.
4. Similarly, P sets the cursor j_1 to the most significant bit of the secret x, and the cursor j_2 to the least significant bit of the secret x.
5. The rapid bit exchange phase starts by V's first challenge sent to P. According to the challenge received by P, c_i', the response r_i' is determined and sent to V and this bit exchange is repeated n times.

Verifier V n-bit secret x		**Prover** P n-bit secret x
Generate $N_a \in_R \{0,1\}^n$		
	N_a →	
		Generate $N_b \in_R \{0,1\}^n$ Compute $R \leftarrow f_x(N_a, N_b)$, $\|R\| = 2n$ $j_1 \leftarrow 1$, $j_2 \leftarrow n$ $k_1 \leftarrow 0$, $k_2 \leftarrow (2n+1)$
	N_b ←	
Compute $R \leftarrow f_x(N_a, N_b)$, $\|R\| = 2n$		
Start of Rapid Bit Exchange		
for $i = 1$ to n		
Pick $c_i \in_R \{0,1\}$		
	c_i c_i'	
Start timer	⟶	

		if $c_i' = 0$
		  if $x_{j_1} = 0$
		  $k_1 \leftarrow (k_1+1)$, $r_i' \leftarrow R_{k_1}$
		  else
		  $k_2 \leftarrow (k_2-2)$, $r_i' \leftarrow R_{k_2}$
		  $j_1 \leftarrow (j_1+1)$
		else
		  if $x_{j_2} = 0$
		  $k_2 \leftarrow (k_2-1)$, $r_i' \leftarrow R_{k_2}$
		  else
		  $k_1 \leftarrow (k_1+2)$, $r_i' \leftarrow R_{k_1}$
		  $j_2 \leftarrow (j_2-1)$

	r_i'' r_i'	
Stop timer	⟵	
Store $r_i'', \triangle t_i$		
End of Rapid Bit Exchange		
	$c_1'\|\|c_2'\|\|...\|\|c_n'$	
	⟵	

if $\sharp(\triangle t_i > \triangle t_{max}) \leq T_t$, $\forall i$
  if $\sharp(c_i \neq c_i') \leq T_c$, $\forall i$
  Compute $(r_i')'s$ using $(c_i')'s$, $\forall i$
  if $\sharp(r_i' \neq r_i'') \leq T_r$, $\forall i$
  ACCEPT		

Fig. 1. Our Protocol

6. In each round, the generated response bit is directly dependent on the challenge received by P and corresponding bit of the secret x. The determination of the response bit is as follows:
   (a) If the challenge bit is 0, the cursor j_1 is activated.
      i. If the bit at that position is 0, the cursor k_1 is incremented and the bit at that position is sent as the response bit.
      ii. Otherwise, the cursor k_2 is decremented twice and the bit at that position is sent as the response bit.
      After one of these two situations is completed, the cursor j_1 is incremented.
   (b) If the challenge bit is 1, the cursor j_2 is activated.
      i. If the bit at that position is 0, the cursor k_2 is decremented and the bit at that position is sent as the response bit.
      ii. Otherwise, the cursor k_1 is incremented twice and the bit at that position is sent as the response bit.
      After one of these two situations is completed, the cursor j_2 is decremented.
7. At the end of each round, V stores the time difference Δt_i and the response received, r_i''.
8. After rapid bit exchange phase is finished, P sends the concatenation of challenges received $(c_1'\|c_2'\|...\|c_n')$ to V by the help of one of the ECC (error correcting code) methods.
9. V checks the number of rounds in which Δt_i exceed Δt_{max} and goes to the next step if this number is smaller than or equals to the value T_t. Otherwise, it ABORTS.
10. V checks the number of rounds in which the challenge sent by V, c_i, is different than the challenge received by P, c_i', and goes to next step if this number is smaller than or equals to the value T_c. Otherwise, it ABORTS.
11. V computes the correct responses r_i' for all challenges received by P, c_i'.
12. Finally, V checks the number of rounds in which the response sent by P, r_i' is different than the response received by V, r_i'', and ACCEPTS if this number is smaller than or equals to the value T_r. Otherwise, it ABORTS.

4 Analysis

In this section, we present the security and performance analyses of our proposed protocol.

4.1 Security Analysis

Secrecy and Non-traceability: In this protocol, if an attacker eavesdrops various sessions, she can only get the nonces N_a and N_b, challenge bits $c_1c_2...c_n$ and response bits $r_1r_2...r_n$. Sending the identity of any party in any step of the protocol is avoided since it violates the privacy as in [20].

Since the values N_a, N_b and $c_1c_2...c_n$ are randomly generated and independent from the secret x, the adversary can not extract x from these values. Besides,

by only observing these values, tracking is not possible since they appear as random in every execution of the protocol. On the other hand, depending on the challenges and the secret x, the response bits $r_1 r_2 ... r_n$ are generated from the output of the pseudo-random keyed function, R. Although an adversary knows the challenges and gets information about n bits of R, she can not have knowledge about the secret x. Similarly, since the response bits $r_1 r_2 ... r_n$ also appear as random, the attacker can not extract the information about the identities of the parties that execute the protocol session. Hence, the attacker will not be able to trace the identities.

False Acceptance Rate (FAR): In order to make the security analysis of this protocol more clear, the fault tolerance concept is ignored. That is, for now, the thresholds T_t, T_c and T_r are assumed to be 0. Clearly, increasing these thresholds with a reasonable size (up to 2-3 errors) will slightly increase the FAR.

In order to estimate the FAR more accurately, the adversary is assumed to generate smart strategies by taking her ability into consideration. There are three strategies that adversaries could follow (according to their capabilities and preferences):

Strategy 1: Initially, the attacker has no information about the secret x and the register R. She carries out the attack as follows:
- Generates random challenges,
- Interrogates the prover with these challenges in order to get the responses corresponding to them,
- Sends these responses to the verifier without any modification.

Strategy 2: The attacker has the capability of learning all bits of the register R and generating the conjectural responses corresponding to the challenges sent by the verifier. The attack can be described as follows:
- Learns the whole register R by the help of the prover,
- Receives the challenge bits sent by the verifier,
  - When the challenge bit is 0, she needs the knowledge of secret x. She does not know the secret x and assumes that it is an alternating series of 0 and 1. Thus, she executes the protocol steps 6.a.i and 6.a.ii described in Section 3, in turn,
  - When the challenge bit is 1, as the same motivation described above holds, she executes the protocol steps 6.b.i and 6.b.ii described in Section 3 one after the other.

Strategy 3: The attacker has the register R and also the secret x, but due to time constraints, she must send the responses to the verifier before the challenges are received. The attacker executes the steps as follows:
- Assumes that the challenge bits form an alternating series of 0 and 1,
- Executes the protocol 6.a and 6.b described in Section 3, in turn.

By only considering the rapid bit exchange phase for any protocol, there is a limit for the FAR as calculated in [2]. In their protocol, the response bit of any step is determined according to not only the present challenge bit, but also all

past challenge bits. Therefore, maximum dependency of the response bit on the challenges is provided. The limit value for FAR is denoted as IC (ideal case) in our graphical results. Due to complexity issues, instead of analytical solution, the simulation results for each strategy described above are generated and compared with the ideal case.

Figure 2 represents the bit-wise success probability of an attacker when she applies one of the strategies explained above. In Figure 2, n increases from 1 to 100. Similarly, Figure 3 demonstrates the overall success probability of an attacker executing one of three strategies and also the ideal case for FAR of any protocol when n increases from 1 to 100. Since these probability values decrease exponentially as n increases, they are calculated and plotted as logarithmic scale.

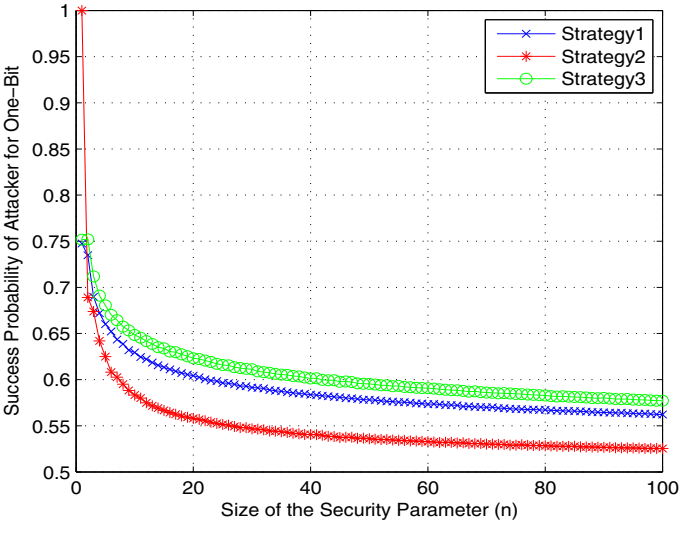


Fig. 2. Success Probability of Attacker for One-Bit

As we can see from the simulation results (see Figure 2 and 3), the success probability of an attacker choosing any of the attack strategies converges to the ideal case. This is achieved by using non-uniform stepping approach in which the register is traversed with non-uniform steps during the determination of the response bit.

By considering these three attack strategies and understanding their estimated FAR's, the choice of the strategies for each type of the frauds can be inferred (see Figure 2 and 3). Distance fraud is realized by utilizing Strategy 3, since the best FAR can be achieved by this strategy and capability of the attacker is sufficient. On the other hand, the mafia and terrorist attackers are not capable of performing Strategy 3, and Strategy 1 provides high FAR compared to Strategy 2, so Strategy 1 is the best for these frauds.

In [2], the ideal case for FAR of any protocol is calculated and also a method that achieves this FAR value is propounded. Although FAR's for distance and

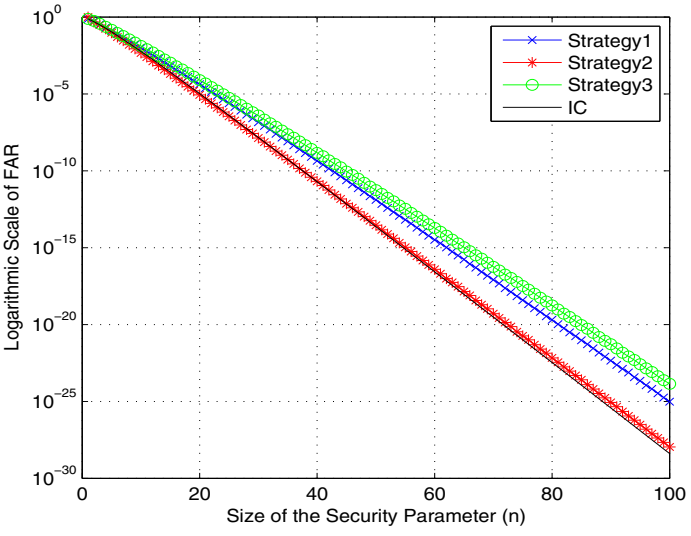


Fig. 3. Logarithmic Scale of FAR

mafia frauds of our protocol are slightly less than those of ideal case, our protocol resists terrorist attacks whereas ideal case does not. Moreover, in our protocol, the noise resilience property is achieved and the large memory requirement of the suggested protocol in [2] is avoided.

Resistance to Timing Attacks: In this protocol, the timing attack can be used as a mean to recover the secret key completely. This can be achieved by utilizing the time difference of preparing response when the secret bit is 0 and 1 can vary.

For example, the attacker acts as a legitimate verifier and sends all challenges zero ($c_i = 0, \forall i$). She measures the differences between the time that the challenge is sent and the time that the response is received. Determining the response bit when the secret bit is 1 takes longer time than determining the response bit when the secret bit is 0. This is because when the secret bit is 0, the cursor is incremented by one, otherwise it is decremented by two. As a result, she can use these time differences in order to recover the secret key.

However, in order to overcome this problem, preparation time of response bits when the secret bit is 0 and 1 can be fixed. That is, the maximum time for preparation of response bit is preferred even if it takes shorter time to calculate the response bit. Therefore, the attacker can not exploit timing difference to recover the secret key.

4.2 Performance Analysis

Computation and Memory Requirement: In this protocol, the amount of computation to be performed by the prover is only the calculation of a pseudo-random function. The rest of the computation involves selection of the appropriate

bits from the output of the pseudo-random function through basic comparison, assignment and increment-decrement operations. Thus, the computational overhead is one hash computation on the prover's side. On the other hand, on the verifier's side, in addition to calculation of pseudo-random function, there are also three operations of tolerance control for the time differences, challenges and responses. This computational workload is assumed to be very weak. Moreover, the signature calculation at the end of the rapid bit exchange phase used in many existing protocols ([3,5,22,23,16,14]) is not performed in this protocol.

The secret shared by both parties is n-bit long. In addition, they compute a $2n$-bit output of pseudo-random function and assign it to $2n$-bit register R. Hence, the total memory requirement on each side is linearly dependent on the security parameter n rather than exponential as in [2].

Noise Resilience: This protocol is tolerant to some delays and faults occurring during transmission. Since the transmission media can have nonlinear characteristics, some responses transmitted from the prover to the verifier can be delayed even if the prover is in the authentication range of the verifier. In order to authenticate the legal prover properly, this delay must be tolerated up to a certain limit, T_t. If the number of time differences that exceed $\triangle t_{max}$ is larger than T_t, the verifier aborts the protocol.

Similarly, since the challenges and the responses may change during the transmission, a tolerance check for them is also performed. As soon as the rapid bit exchange phase is finished, the prover sends the concatenation of the challenges received $c'_1||c'_2||...||c'_n$ to the verifier by the help of one of the ECC (error correcting code) method. The verifier computes the number of differences between the challenges sent by itself (c_i) and the challenges received by the prover (c'_i). If this number is larger than T_c, the verifier aborts the session. Otherwise, she computes the responses (r'_i) corresponding to the challenges sent by the prover (c'_i). The verifier counts the number of differences between the responses computed (r'_i) and the responses received by itself (r''_i) to decide whether authentication is successful or not. If this number is larger than T_r, she aborts the protocol or else authenticates the prover.

5 Protocol Comparison

In this section, characteristics of the previous protocols and our protocol are summarized. The resistance to distance, mafia and terrorist frauds, computational overhead, usage of final signature and noise resilience are the properties that we use for comparison. The differences in these features between the previous protocols and our proposed protocol can be clearly seen in Table 2.

If solely the terrorist fraud is taken into consideration, the resistance to this attack is provided in the protocols described in [20], [23] and [14] and our proposed protocol. Furthermore, the best result is obtained with our protocol. Similarly, when the success probability of distance and mafia frauds are considered together, among all protocols that have no final signature, the ones less vulnerable to these attacks are [2] and our protocol. However, the protocol in [2] does

Table 2. Comparison of Distance Bounding Protocols

	Distance fraud	Mafia fraud	Terrorist fraud	Computational overhead	Final Signature	Noise Resilience
BC [3]	$(1/2)^n$	$(1/2)^n$	1	2	Yes	No
Capkun et al. [5]	$(1/2)^n$	$(1/2)^n$	1	4	Yes	No
HK [10]	$(3/4)^n$	$(3/4)^n$	1	1	No	Yes
Reid et al. [20]	$(3/4)^n$	$(3/4)^n$	$(3/4)^n$	1	No	Yes
SP [22]	$(1/2)^n$	$(1/2)^n$	1	4	Yes	Yes
TP [23]	$(3/4)^n$	$(9/16)^n$	$(3/4)^n$	5	Yes	No
MP [16]	$(3/4)^n$	$(3/5)^n$	1	1	No	Yes
Kim et al. [14]	$(3/4)^n$	$(1/2)^n$	$(3/4)^n$	2	Yes	Yes
AT [2]	$> (1/2)^n$	$> (1/2)^n$	1	1	No	No
KA [13]	$(7/8)^n$	$(1/2)^n$	1	1	No	Yes
Our scheme	$> (1/2)^n$	$> (1/2)^n$	$> (1/2)^n$	1	No	Yes

not ensure protection against terrorist fraud and errors/delays stemmed from communication channel. In addition, as stated in Section 4, the large memory requirement in [2] is avoided in our proposed protocol.

6 Conclusion

In this paper, we present a new secure distance-bounding protocol for RFID systems that significantly enhances the already proposed ones. It uses non-uniform stepping approach in which binary responses are created based on the secret key and binary challenges. The most important contribution of this paper is that our protocol converges the false acceptance rate to the ideal case $(1/2)^n$ in the presence of all relay attacks. Therefore, it is the first one that provides all security features at once among already proposed protocols that does not need final signature. Our protocol also has noise resilience property and provides the best performance in terms of memory and computation when taking into consideration the security performance that it provides.

Acknowledgments

The authors thank to Mahmut Şamil Sağiroğlu for his interesting comments.

References

1. Avoine, G., Bingöl, M.A., Kardaş, S., Lauradoux, C., Martin, B.: A Framework for Analyzing RFID Distance Bounding Protocols. Journal of Computer Security – Special Issue on RFID System Security (2010)
2. Avoine, G., Tchamkerten, A.: An efficient distance bounding RFID authentication protocol: balancing false-acceptance rate and memory requirement. In: Samarati, P., Yung, M., Martinelli, F., Ardagna, C.A. (eds.) ISC 2009. LNCS, vol. 5735, pp. 250–261. Springer, Heidelberg (2009)

3. Brands, S., Chaum, D.: Distance Bounding Protocols (Extended Abstract). In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 344–359. Springer, Heidelberg (1994)
4. Bussard, L.: Trust Establishment Protocols for Communicating Devices. PhD thesis, Eurecom-ENST (Paris, France)
5. Capkun, S., Buttyán, L., Hubaux, J.P.: SECTOR: secure tracking of node encounters in multi-hop wireless networks. In: SASN, pp. 21–32 (2003)
6. Desmedt, Y.G., Goutier, C., Bengio, S.: Special Uses and Abuses of the Fiat Shamir Passport Protocol. In: Pomerance, C. (ed.) CRYPTO 1987. LNCS, vol. 293, pp. 21–39. Springer, Heidelberg (1988)
7. Drimer, S., Murdoch, S.: Keep Your Enemies Close: Distance Bounding Against Smartcard Relay Attacks. In: Proceedings of USENIX Security (2007)
8. Finkenzeller, K.: RFID Handbook. John Wiley and Sons, Chichester (2003)
9. Hancke, G.: A Practical Relay Attack on ISO 14443 Proximity Cards (2005) (manuscript)
10. Hancke, G., Kuhn, M.: An RFID Distance Bounding Protocol. In: Conference on Security and Privacy for Emerging Areas in Communication Networks – SecureComm 2005, Athens, Greece, IEEE, pp. 67–73. IEEE Computer Society, Los Alamitos (2005)
11. Kara, O., Kardaş, S., Bingöl, M.A., Avoine, G.: Optimal Security Limits of RFID Distance Bounding Protocols. In: Ors Yalcin, S.B. (ed.) RFIDSec 2010. LNCS, vol. 6370, pp. 220–238. Springer, Heidelberg (2010)
12. Kfir, Z., Wool, A.: Picking Virtual Pockets Using Relay Attacks on Contactless Smartcard Systems. In: Conference on Security and Privacy for Emerging Areas in Communication Networks – SecureComm 2005, Athens, Greece, IEEE, pp. 67–73. IEEE Computer Society, Los Alamitos (2005)
13. Kim, C.H., Avoine, G.: RFID Distance Bounding Protocol with Mixed Challenges to Prevent Relay Attacks. In: Garay, J.A., Miyaji, A., Otsuka, A. (eds.) CANS 2009. LNCS, vol. 5888, pp. 119–133. Springer, Heidelberg (2009)
14. Kim, C.H., Avoine, G., Koeune, F., Standaert, F.X., Pereira, O.: The Swiss-Knife RFID Distance Bounding Protocol. In: Lee, P., Cheon, J. (eds.) ICISC 2008. LNCS, vol. 5461, pp. 98–115. Springer, Heidelberg (2009)
15. Melia-Segui, J., Garcia-Alfaro, J., Herrera-Joancomarti, J.: Analysis and Improvement of a Pseudorandom Number Generator for EPC Gen2 Tags. In: Sion, R., Curtmola, R., Dietrich, S., Kiayias, A., Miret, J.M., Sako, K., Sebé, F. (eds.) RLCPS, WECSR, and WLC 2010. LNCS, vol. 6054, pp. 34–46. Springer, Heidelberg (2010)
16. Munilla, J., Peinado, A.: Distance Bounding Protocols for RFID Enhanced by using Void-Challenges and Analysis in Noisy Channels. Wireless Communications and Mobile Computing 8, 1227–1232 (2008)
17. Munilla, J., Peinado, A.: Attacks on a distance bounding protocol. Computer Communications 33, 884–889 (2010)
18. Nikov, V., Vauclair, M.: Yet Another Secure Distance-Bounding Protocol. Cryptology ePrint Archive, Report 2008/319 (2008), http://eprint.iacr.org/
19. Peris-Lopez, P., Hernandez-Castro, J.C., Estevez-Tapiador, J.M., van der Lubbe, J.C.A.: Shedding Some Light on RFID Distance Bounding Protocols and Terrorist Attacks. arXiv.org, Computer Science, Cryptography and Security (2009)
20. Reid, J., Gonzalez Neito, J., Tang, T., Senadji, B.: Detecting Relay Attacks with Timing Based Protocols. In: Bao, F., Miller, S. (eds.) Proceedings of the 2nd ACM Symposium on Information, Computer and Communications Security – ASIACCS 2007, Singapore, Republic of Singapore, pp. 204–213. ACM, New York (2007)

21. Singelée, D., Preneel, B.: Location verification using secure distance bounding protocols. In: IEEE International Conference on Mobile Adhoc and Sensor Systems Conference, p. 840 (2005)
22. Singelée, D., Preneel, B.: Distance Bounding in Noisy Environments. In: Stajano, F., Meadows, C., Capkun, S., Moore, T. (eds.) ESAS 2007. LNCS, vol. 4572, pp. 101–115. Springer, Heidelberg (2007)
23. Tu, Y.J., Piramuthu, S.: RFID Distance Bounding Protocols. In: First International EURASIP Workshop on RFID Technology, Vienna, Austria (2007)

E-Ticketing Scheme for Mobile Devices with Exculpability

Arnau Vives-Guasch[1], Magdalena Payeras-Capella[2], Macià Mut-Puigserver[2], and Jordi Castellà-Roca[1]

[1] Dpt. d'Enginyeria Informàtica i Matemàtiques, UNESCO Chair in Data Privacy, Universitat Rovira i Virgili, Av. Països Catalans 26, E-43007 Tarragona, Spain
{arnau.vives,jordi.castella}@urv.cat

[2] Dpt. de Ciències Matemàtiques i Informàtica, Universitat de les Illes Balears, Ctra. de Valldemossa, km 7,5. 07120 Palma de Mallorca, Spain
{mpayeras,macia.mut}@uib.es

Abstract. An electronic ticket is a contract, in digital format, between the user and the service provider, and reduces both economic costs and time in many services such as air travel industries or public transport. However, the electronic ticket security has to be strongly guaranteed, as well as the privacy of their users. We present an electronic ticketing system that considers these security requirements and includes the exculpability as a security requirement for these systems, i.e users and the service provider can not falsely accuse each other of misbehaviour. The system ensures that either both parties receive their desired data from other or neither does (fair exchange). Furthermore, this scheme takes special care of the computational requirements on the users' side, as we consider the use of mobile devices with light-weight cryptography, because they are the best positioned in order to manage the electronic tickets in the near future.

Keywords: e-ticketing, e-commerce, security, privacy, exculpability.

1 Introduction

Information technologies (IT) are becoming usual in our society as they progressively replace the use of paper in many of our common operations. An example of paper ticket could be the air flight boarding pass. Vodafone and Spanair[1] conducted an e-ticketing test in Spain (May 2007). The passengers received the electronic boarding pass in their mobile phone, and they were able to go directly to the security control area, and later board. The International Air Transport Association (IATA) started in 2004 a program to introduce the use of electronic tickets. IATA estimates that the no-use of paper tickets will reduce the costs by US$ 3000M[2], boosting disintermediation by using electronic tickets. The electronic tickets have not been used only as a boarding pass. They can also be used

[1] http://www.spanair.com/web/es-es/Sobre-Spanair/Noticias-y-eventos/Spanair-y-Vodafone-Espana-presentan-la-tarjeta-de-embarque-movil/

[2] IATA:http://www.iata.org/pressroom/pr/2008-31-05-01.htm

J. Garcia-Alfaro et al. (Eds.): DPM 2010 and SETOP 2010, LNCS 6514, pp. 79–92, 2011.

in multiple transport services. The AMSBUS[3] booking system from the Czech Republic allows the purchase of SMS tickets. The passenger receives the ticket in her mobile phone; then, the user shows the message to the ticket inspector when needed. Leeds United[4] supporters can book one of their sport events and later receive an SMS with the booking confirmation together with some added information such as the assigned seat.

These examples show the progressive introduction of electronic tickets on different kinds of services and the increasing use of mobile phones in all of them as the most suitable e-ticket storage device. In addition to that, the real application of these electronic ticketing systems depends on their security, due to the ease of copy of electronic data. Electronic tickets have to keep the same security that is offered in paper tickets.

1.1 Organization

An analysis of the previous works related to our proposal is presented in Section 2. Later, in Section 3, our scheme is accurately defined. The security and privacy analysis of the scheme is detailed in Section 4. Finally, the conclusions and future work are described in Section 5.

2 Previous Works

First of all, in Section 2.1, we start presenting the security requirements that have to be achieved in these systems, as well as we introduce a new security requirement that is not achieved in the previous works and which could be taken into account in the future: *exculpability*. Once the security requirements are described, the e-ticketing proposals have been classified in Section 2.2.

2.1 Security Requirements

The e-ticketing systems have to consider and guarantee the following security requirements:

- Authenticity : e-tickets should take measures to avoid falsification.
- Non-repudiation: the issuer can not deny the emission of an authentic ticket.
- Integrity: an issued ticket can not be further modified by anyone.
- Anonymity: there are two anonymity degrees: non-anonymous (the service requires user identification and authentication) and revocable anonymous (the service is anonymous, but it can be revoked if the user misbehaves).
- Reusability: a ticket could be used once (non-reusable) or many times (reusable). In both cases, ticket overspending has to be prevented.
- Expiry date: a ticket could be only valid during a time interval.
- Online/Offline: ticket verification can require a persistent connection with a trusted centralized system (online); otherwise, that connection is never necessary (offline).

[3] AMSBUS: `http://www.svt.cz/en/amsbus/`
[4] Leeds United: `http://www.leedsunited.com/`

Exculpability. None of the analyzed proposals deals with exculpability; that is, the service provider can not falsely accuse the user of ticket overspending, and the user is able to demonstrate that she has already validated the ticket before using it. The exculpability is an interesting issue in our case, because the e-ticketing scheme should ensure that either both parties (users and provider) receive their desired data (e-ticket and the validated e-ticket) from other or neither does (fair exchange). The parties agree to reveal its data only if the other part also agrees. If any party deviates from the scheme then it can be identified as the culprit by the Trusted Third Party (TTP). Our scheme defined in Section 3 takes exculpability as a security requirement for an e-ticketing system, as a first step to include this security requirement in future works.

2.2 Classification of Proposals

In this section, the proposals have been differentiated by the devices used in the systems: the *smart-card-based*, and the *non-smart-card-based* ones, with a deeper analysis performed in the non-smart-card-based proposals, taking into account their anonymity compliance.

Smart-card-based. Smart-card-based proposals [5,7,13,12,14,19,18] establish a communication channel with the verification system. Thus, the most sensitive operations are sent to the smart-card through this channel. The smart-card verifies each operation, so that users can not perform any non-allowed action. Security of smart-card-based systems rely on the smart-card security. If the smart-card security is compromised, the security of the entire system is also compromised. One recent example is the case of the Mifare cards, where in [10] the authors succeeded to compromise them. As a result, all the entire public transport system that used exclusively Mifare cards was compromised.

Non-smart-card-based. Non-smart-card-based systems [15,6,1,8,16,9,3] allow to perform applications with higher computation requirements while taking advantage of their high storage capacity and also their wireless short-range communication resources; this is the case of the mobile phones, smart phones or PDA's. However, as these devices are not considered tamper-proof devices, e-ticketing systems require then high-level cryptographic protection in order to assure that users follow the e-ticketing protocol correctly. We classify the non-smart-card-based systems depending on *non-anonymous* and *revocable-anonymous* compliance.

Non-Anonymous proposals. Some proposals are oriented to services where anonymity can not be provided to the user, or simply, these systems are not conceived to achieve anonymity at all. Between the proposals that do not consider anonymity, digital signatures are commonly used [6,1].

Revocable-Anonymous proposals. In the analyzed proposals, digital signatures are commonly used [15,8,16,9,3], providing ticket authenticity, non-repudiation

and integrity. Most of them are based on Chaum's blind signatures [2] with the use of pseudonyms for revocable anonymity. There is a diversity of considerations in the ticket reusability; Patel and Crowcroft [15] believe that tickets can be reusable; Haneberg et al. [8], Heydt-Benjamin et al. [9] and Chen et al. [3] do not consider ticket reusability; finally, Quercia and Hailes [16] consider that reusability mainly depends on the service. There is a remarkable equality between the proposals that use online verification [15,9] and the ones which use offline verification [8,16,3].

3 E-Ticketing Scheme

The scheme has the following actors: the user $\mathcal{U}$; the ticket issuer $\mathcal{I}$, who sends a valid ticket to $\mathcal{U}$; the service provider $\mathcal{P}$, who verifies the ticket and gives the service; and finally the TTP $\mathcal{T}$, who preserves $\mathcal{U}$'s anonymity, and gives a valid non-identity pseudonym to $\mathcal{U}$. The e-ticketing scheme has been designed for mobile devices, reducing the computation requirements in the user side, and providing the basic security requirements (authenticity, non-repudiation and integrity) together with expiry date, revocable anonymity and exculpability. The ticket verification is performed offline when there is only one provider for each service, although the scheme could be extended with multiple providers that offer the same service. In that extended case, the scheme would prevent overspending through online verification, requiring then the interconnection of the service providers which offer the same service. In any case, the connection with the issuer is never necessary. In Table 1, we define the details of our proposal, as well as some notation that is used in the scheme.

Table 1. Details of the proposal: security requirements, actors, ticket information and receipt information

Security Requirements			
Authenticity		Non-repudiation	
Integrity		Revocable anonymity	
Non-overspending		Offline verification	
Expiry date		Exculpability	
Actors			
User	$\mathcal{U}$	Service Provider	$\mathcal{P}$
Ticket Issuer	$\mathcal{I}$	Trusted Third Party	$\mathcal{T}$
Ticket Information (T)			
Serial number	Sn	Issuer	Is
Service	Sv	Terms and conditions	Tc
User pseudonym	$\mathsf{Pseu}_{\mathcal{U}}$	Attributes	At
Type of ticket	Ty	Encrypted verification data	$\delta_{\mathcal{I},\mathcal{P}}$
Validity time	Tv	Date of issue	Ti
Exculpability ($\mathcal{U}$)	$\mathsf{h}_{r_{\mathcal{U}}}$	Exculpability ($\mathcal{P}$)	$\mathsf{h}_{r_{\mathcal{I}}}$
Digital signature of $\mathcal{I}$	$\mathsf{Sign}_{\mathcal{I}}(\mathsf{T})$		
Receipt Information (R)			
Encrypted exculpability ($\mathcal{P}$)	$\mathsf{A}_{\mathcal{P}}$	Timestamp	τ_i
Ticket serial number	$\mathsf{T.Sn}$	Digital signature of $\mathcal{P}$	$\mathsf{Sign}_{\mathcal{P}}(\mathsf{R})$

3.1 Phases

The phases of our system are: *Pseudonym Renewal*, where the user obtains a new temporal pseudonym to be used in the system without linkage to user's identity (if user behaves correctly); *Ticket Purchase*, that consists on the payment of the service and reception of the ticket; and *Ticket Verification*, where the user shows the ticket to the service provider in order to be checked and validated. Other phases considered in the system are claims. These claims should only be executed in case of controversial situations during the *Ticket Verification* phase: *Claim* m_2 *Not Received* (when $\mathcal{U}$ sends the first step of the verification m_1 but does not receive m_2 by $\mathcal{P}$, or the information is not correct); *Claim* m_3 *Not Received* (when $\mathcal{P}$ sends the second step of the verification m_2 but does not receive m_3 by $\mathcal{U}$, or the information is not correct); and *Claim* m_4 *Not Received* (when $\mathcal{U}$ sends the third step of the verification m_3 but does not receive m_4 by $\mathcal{P}$, or the information is not correct). Users have a digital credential ($\mathsf{Cert}_{\mathcal{U}}$) for authentication to the TTP only, as the system is anonymous, and all further movements in the system are tracked only with the assigned temporal pseudonym ($\mathsf{Pseu}_{\mathcal{U}}$).

Pseudonym Renewal. The user $\mathcal{U}$ contacts the pseudonym manager $\mathcal{T}$ in order to renew the assigned pseudonym. The certificate $\mathsf{Cert}_{\mathcal{U}}$ identifies $\mathcal{U}$ through a secure connection established between the two parties. The system has the public parameters (α, p, q), where α is a generator of the group G with order p, being p and q large primes achieving $p = 2q + 1$. $\mathcal{U}$ generates a random value $\mathsf{x}_{\mathcal{U}} \overset{R}{\leftarrow} \mathbb{Z}_q$ and computes $\mathsf{y}_{\mathcal{U}} = \alpha^{\mathsf{x}_{\mathcal{U}}} \pmod p$ in order to receive a valid signed pseudonym $\mathsf{Pseu}_{\mathcal{U}}$ from $\mathcal{T}$. $\mathcal{U}$ and $\mathcal{T}$ have their own pair of keys used for signature and encryption of the transmitted data between them.

authenticateUser User $\mathcal{U}$ follows the next steps:

1. generates $\mathsf{x}_{\mathcal{U}} \overset{R}{\leftarrow} \mathbb{Z}_q$, and computes $\mathsf{y}_{\mathcal{U}} = \alpha^{\mathsf{x}_{\mathcal{U}}} \pmod p$;
2. computes the signature $\mathsf{sk}_{\mathcal{U}}(\mathsf{h}_{\mathsf{y}_{\mathcal{U}}})$ [5] where $\mathsf{h}_{\mathsf{y}_{\mathcal{U}}} = hash(\mathsf{y}_{\mathcal{U}})$ [6];
3. encrypts $\mathsf{y}_{\mathcal{U}}$ with the $\mathcal{T}$'s public key: $\mathsf{pk}_{\mathcal{T}}(\mathsf{y}_{\mathcal{U}})$ [7];
4. sends $(\mathsf{sk}_{\mathcal{U}}(\mathsf{h}_{\mathsf{y}_{\mathcal{U}}}), \mathsf{Cert}_{\mathcal{U}}, \mathsf{pk}_{\mathcal{T}}(\mathsf{y}_{\mathcal{U}}))$ to $\mathcal{T}$;

generatePseudonym Pseudonym Manager $\mathcal{T}$ executes:

1. decrypts $\mathsf{sk}_{\mathcal{T}}(\mathsf{pk}_{\mathcal{T}}(\mathsf{y}_{\mathcal{U}})) \rightarrow (\mathsf{y}_{\mathcal{U}})$;
2. verifies $\mathsf{y}_{\mathcal{U}}$: $\mathsf{pk}_{\mathcal{U}}(\mathsf{sk}_{\mathcal{U}}(\mathsf{h}_{\mathsf{y}_{\mathcal{U}}})) \rightarrow (\mathsf{h}_{\mathsf{y}_{\mathcal{U}}}) \overset{?}{=} hash(\mathsf{y}_{\mathcal{U}})$;
3. if correct, then computes the signature of $\mathsf{sk}_{\mathcal{T}}(\mathsf{h}_{\mathsf{y}_{\mathcal{U}}})$; and
4. sends $\mathsf{Pseu}_{\mathcal{U}} = (\mathsf{y}_{\mathcal{U}}, \mathsf{sk}_{\mathcal{T}}(\mathsf{h}_{\mathsf{y}_{\mathcal{U}}}))$ to $\mathcal{U}$.

verifyPseudonym $\mathcal{U}$ computes:

1. verifies $\mathsf{y}_{\mathcal{U}}$: $\mathsf{pk}_{\mathcal{T}}(\mathsf{sk}_{\mathcal{T}}(\mathsf{h}_{\mathsf{y}_{\mathcal{U}}})) \rightarrow (\mathsf{h}_{\mathsf{y}_{\mathcal{U}}}) \overset{?}{=} hash(\mathsf{y}_{\mathcal{U}})$;

[5] Note that $\mathsf{sk}_{\mathcal{E}}(content)$ means the decryption of *content* or the generation of a signature with its content *content* by using the private key of the entity $\mathcal{E}$.

[6] Note that $hash()$ is a public cryptographic one-way summarizing function that achieves collision-resistance.

[7] Note that $\mathsf{pk}_{\mathcal{E}}(content)$ means the encryption of *content* or the verification of a signature *content* by using the public key of the entity $\mathcal{E}$.

Ticket Purchase. The user establishes a connection with the ticket issuer $\mathcal{I}$ in order to receive the ticket. This connection could be established through an anonymous channel like TOR [4], guaranteeing then user's privacy. There are current contributions [8] that have implemented TOR for mobile devices with Android. $\mathcal{I}$ has a key pair and its public key certificate ($\mathsf{Cert}_{\mathcal{I}}$). Users do not use their personal keys (it would cause loss of anonymity); they use the temporal pseudonyms and authenticate through the Schnorr's Zero-Knowledge Proof (ZKP) [17]. The payment method is considered as out of scope in this proposal as we focus on the privacy given to user when joining/exiting the system, and using the service.

$\mathcal{I}$ generates the ticket with the information and its digital signature, together with the secret value $\mathsf{r}_{\mathcal{I}}$ and the secret shared key (they are decryptable only by $\mathcal{P}$ and $\mathcal{T}$) in order to let the provider show the secret value $\mathsf{r}_{\mathcal{I}}$ later, in the verification phase. The ticket issuer $\mathcal{I}$ and the user $\mathcal{U}$ follow this protocol:

getService $\mathcal{U}$ executes:

1. selects and pays for the desired service Sv;
2. generates a random value $\mathsf{r}_{\mathcal{U}} \overset{R}{\leftarrow} \mathbb{Z}_q$, and computes $\mathsf{h}_{\mathsf{r}_{\mathcal{U}}} = hash(\mathsf{r}_{\mathcal{U}})$;
3. computes $\mathsf{H}_{\mathcal{U}} = \alpha^{r_{\mathcal{U}}} \pmod p$;
4. generates two more random values $a_1, a_2 \overset{R}{\leftarrow} \mathbb{Z}_q$ to be used in the Schnorr proof;
5. computes $A_1 = \alpha^{a_1} \pmod p$;
6. computes $A_2 = \alpha^{a_2} \pmod p$;
7. sends $(\mathsf{Pseu}_{\mathcal{U}}, \mathsf{H}_{\mathcal{U}}, A_1, A_2, \mathsf{h}_{\mathsf{r}_{\mathcal{U}}}, \mathsf{Sv})$ to the ticket issuer $\mathcal{I}$.

getChallenge $\mathcal{I}$ follows the next steps:

1. generates and sends a challenge $\mathsf{c} \overset{R}{\leftarrow} \mathbb{Z}_q$ for $\mathcal{U}$;
2. asynchronously, for optimization, pre-computes $\mathsf{y}_{\mathcal{U}}{}^{\mathsf{c}} \pmod p$;
3. asynchronously, for optimization, pre-computes $\mathsf{H}_{\mathcal{U}}{}^{\mathsf{c}} \pmod p$;

solveChallenge $\mathcal{U}$ computes:

1. computes $\mathsf{w}_1 = a_1 + \mathsf{c} \cdot \mathsf{x}_{\mathcal{U}} \pmod q$;
2. computes $\mathsf{w}_2 = a_2 + \mathsf{c} \cdot \mathsf{r}_{\mathcal{U}} \pmod q$;
3. encrypts $(\mathsf{w}_1, \mathsf{w}_2)$ and sends it to $\mathcal{I}$: $\mathsf{pk}_{\mathcal{I}}((\mathsf{w}_1, \mathsf{w}_2))$;
4. pre-computes the shared session key used in the ticket verification: $\mathsf{K} = hash(\mathsf{w}_2)$;

getTicket $\mathcal{I}$ follows the next steps:

1. decrypts $\mathsf{sk}_{\mathcal{I}}(\mathsf{pk}_{\mathcal{I}}(\mathsf{w}_1, \mathsf{w}_2)) \rightarrow (\mathsf{w}_1, \mathsf{w}_2)$;
2. computes $\alpha^{\mathsf{w}_1} \pmod p$;
3. computes $\alpha^{\mathsf{w}_2} \pmod p$;
4. verifies $\alpha^{\mathsf{w}_1} \overset{?}{=} A_1 \cdot \mathsf{y}_{\mathcal{U}}{}^{\mathsf{c}} \pmod p$;
5. verifies $\alpha^{\mathsf{w}_2} \overset{?}{=} A_2 \cdot \mathsf{H}_{\mathcal{U}}{}^{\mathsf{c}} \pmod p$;
6. computes the shared session key: $\mathsf{K} = hash(\mathsf{w}_2)$;
7. obtains a unique serial number Sn, and a random value $\mathsf{r}_{\mathcal{I}} \overset{R}{\leftarrow} \mathbb{Z}_p$;
8. computes $\mathsf{h}_{\mathsf{r}_{\mathcal{I}}} = hash(\mathsf{r}_{\mathcal{I}})$;
9. composes $\kappa = (\mathsf{K}, \mathsf{r}_{\mathcal{I}})$ and signs it $\kappa^* = (\kappa, \mathsf{Sign}_{\mathcal{I}}(\kappa))$;
10. encrypts κ^* with a digital envelope which is decryptable by the TTP $\mathcal{T}$ and the provider $\mathcal{P}$ for possible future controversial situations during the ticket verification: $\delta_{\mathcal{T},\mathcal{P}} = \mathsf{pk}_{\mathcal{T},\mathcal{P}}(\kappa^*)$. If $\mathcal{I}$ tries to falsify $\delta_{\mathcal{T},\mathcal{P}}$, the $\mathcal{T}$ can check that information and, demonstrate that $\mathcal{I}$ is the culprit;

[8] `http://sourceforge.net/apps/trac/silvertunnel/wiki/TorJavaOverview`

11. fills out the ticket information T (Sn, Sv, $\mathsf{Pseu}_{\mathcal{U}}$, Tv, Ti, $\mathsf{h}_{r_{\mathcal{I}}}$, $\mathsf{h}_{r_{\mathcal{U}}}$, $\delta_{\mathcal{T},\mathcal{P}}$, etc.);
12. digitally signs the ticket T, and obtains the signed ticket, $\mathsf{Sign}_{\mathcal{I}}(\mathsf{T}) = \mathsf{sk}_{\mathcal{I}}(hash(\mathsf{T}))$, and $\mathsf{T}^* = (\mathsf{T}, \mathsf{Sign}_{\mathcal{I}}(\mathsf{T}))$;
13. sends T^* to the user $\mathcal{U}$.

receiveTicket $\mathcal{U}$ executes:

1. verifies the digital signature $\mathsf{Sign}_{\mathcal{I}}(\mathsf{T})$ of the ticket T using the issuer's certificate;
2. verifies that ticket T data and the performed request match;
3. verifies the ticket validity ($\mathsf{T.Ti}, \mathsf{T.Tv}$);
4. verifies $\mathsf{T.Pseu}_{\mathcal{U}}$;
5. stores $(\mathsf{T}^*, \mathsf{r}_{\mathcal{U}})$ in the device.

Ticket Verification. When the user wants to use the service, she must verify the ticket in advance. For simplicity, we present the ticket verification with only one provider, so the service provider $\mathcal{P}$ never needs permanent communication with the ticket issuer. Nonetheless, the protocol can be extended for multiple providers. In that case, all the service providers should be connected to a central repository of spent tickets in order to control ticket overspending. The user only interacts with the service provider, but in controversial situations, she and/or the service provider could interact directly with the TTP through a resilient connection in order to preserve the security requirements of the protocol. If user misbehaved, her identity could be revoked, enabling to take further actions.

$\mathcal{U}$ sends the ticket T^*, and $\mathcal{P}$ checks it. If passed, $\mathcal{P}$ sends the commitment so that $\mathsf{r}_{\mathcal{I}}$ will be disclosed if $\mathcal{U}$ behaves correctly. Once the user sends the secret value $\mathsf{r}_{\mathcal{U}}$ encrypted through a shared key, then she receives the secret $\mathsf{r}_{\mathcal{I}}$ together with the receipt R^* from $\mathcal{P}$. The service provider $\mathcal{P}$ and the user $\mathcal{U}$ do the following steps:

showTicket $\mathcal{U}$ computes:

1. sends ticket $\mathsf{m}_1 = \mathsf{T}^*$ to $\mathcal{P}$;

verifyTicket $\mathcal{P}$ executes:

1. verifies the ticket signature, $\mathsf{T.Sv}$, $\mathsf{T.Ti}$, and $\mathsf{T.Tv}$;
2. if the verifications fail, $\mathcal{P}$ omits m_1, and aborts the ticket verification;
3. else $\mathcal{P}$ looks for the ticket T^* in the database using $\mathsf{T.Sn}$; and verifies that the ticket has not been spent: $\stackrel{?}{\exists}\, \mathsf{r}_{\mathcal{U}}$ linked to T^* in the DB;
   (a) if $\nexists \mathsf{r}_{\mathcal{U}}$ linked to T^*:
      i. computes $\mathsf{A}_{\mathcal{P}} = PRNG(\mathsf{h}_{\mathsf{K}}) \oplus \mathsf{r}_{\mathcal{I}}$, where $PRNG(\mathsf{h}_{\mathsf{K}})$ is a secure pseudorandom number generator and, $\mathsf{h}_{\mathsf{K}} = hash(\mathsf{K})$ is the seed. Note that K and $\mathsf{r}_{\mathcal{I}}$ are obtained from $\delta_{\mathcal{T},\mathcal{P}}$;
      ii. encrypts $\mathsf{A}_{\mathcal{P}}$ with the public key of the TTP $\mathcal{T}$: $\mathsf{pk}_{\mathcal{T}}(\mathsf{A}_{\mathcal{P}})$;
      iii. assigns $\mathsf{V}_{\mathsf{succ}} = (\mathsf{T.Sn}, \mathsf{flag}_1, \tau_1, \mathsf{pk}_{\mathcal{T}}(\mathsf{A}_{\mathcal{P}}))$, ($\tau_1$ is the verification timestamp). The flag flag_1 indicates that the ticket is valid and has not been spent yet. The signature is noted: ${\mathsf{V}_{\mathsf{succ}}}^* = (\mathsf{V}_{\mathsf{succ}}, \mathsf{sk}_{\mathcal{P}}(hash(\mathsf{V}_{\mathsf{succ}})))$;
      iv. sends $\mathsf{m}_2 = {\mathsf{V}_{\mathsf{succ}}}^*$ to $\mathcal{U}$;
   (b) if $\exists \mathsf{r}_{\mathcal{U}}$ linked to T^*:
      i. assigns $\mathsf{V}_{\mathsf{fail}} = (\mathsf{T.Sn}, \mathsf{r}_{\mathcal{U}}, \mathsf{flag}_0, \tau_1)$. The flag_0 indicates that the ticket has been spent, i.e. is not valid. The signature is noted: ${\mathsf{V}_{\mathsf{fail}}}^* = (\mathsf{V}_{\mathsf{fail}}, \mathsf{sk}_{\mathcal{P}}(hash(\mathsf{V}_{\mathsf{fail}})))$;
      ii. sends $\mathsf{m}_2 = {\mathsf{V}_{\mathsf{fail}}}^*$ to $\mathcal{U}$;

showProof $\mathcal{U}$ executes:

1. verifies $\mathcal{P}$'s signature;
2. if $\mathsf{V_{succ}}^*$ or either $\mathsf{V_{fail}}^*$ are not received, the *Claim* m_2 *Not Received* is called;
   (a) if $\mathsf{V_{fail}}^*$ is received, $\mathcal{U}$ aborts the verification process. If the response is not correct, $\mathcal{U}$ can contact with the TTP to reconsider the situation by calling *Claim* m_2 *Not Received*;
   (b) if $\mathsf{m}_2 = \mathsf{V_{succ}}^*$ is received, $\mathcal{U}$ has to verify the signature and data. If verifications are correct she continues the protocol. Otherwise, $\mathcal{U}$ can contact the TTP by calling *Claim* m_2 *Not Received*;
3. calculates $\mathsf{A}_\mathcal{U} = PRNG(\mathsf{K}) \oplus \mathsf{r}_\mathcal{U}$, using the shared value K as seed;
4. sends $\mathsf{m}_3 = (\mathsf{T.Sn}, \mathsf{A}_\mathcal{U})$ to $\mathcal{P}$;

verifyProof $\mathcal{P}$ follows the next steps:

1. obtains $\mathsf{T.Sn}$, and computes $\mathsf{r}_\mathcal{U} = \mathsf{A}_\mathcal{U} \oplus PRNG(\mathsf{K})$;
2. verifies $\mathsf{h}_{\mathsf{r}_\mathcal{U}} \stackrel{?}{=} hash(\mathsf{r}_\mathcal{U})$;
3. if $\mathsf{r}_\mathcal{U}$ is not received or does not match, the *Claim* m_3 *Not Received* is called;
4. generates τ_2 and verifies it using the ticket expiry date $(\mathsf{T.Ti}, \mathsf{T.Tv})$ and the timestamp τ_1;
5. signs $\mathsf{A}_\mathcal{P}$ approving then the validation with timestamp τ_2: $\mathsf{R} = (\mathsf{A}_\mathcal{P}, \mathsf{T.Sn}, \tau_2)$, and $\mathsf{R}^* = (\mathsf{R}, \mathsf{sk}_\mathcal{P}(hash(\mathsf{R})))$;
6. sends $\mathsf{m}_4 = \mathsf{R}^*$ to $\mathcal{U}$;

getValidationConfirmation $\mathcal{U}$ follows the next steps:

1. checks the signature of R^*;
2. if $\mathsf{r}_\mathcal{I}$ is not received or is incorrect, the *Claim* m_4 *Not Received* is called;
3. computes $\mathsf{r}_\mathcal{I} = \mathsf{A}_\mathcal{P} \oplus PRNG(\mathsf{h_K})$;
4. verifies $\mathsf{h}_{\mathsf{r}_\mathcal{I}} \stackrel{?}{=} hash(\mathsf{r}_\mathcal{I})$;
5. if all verifications are correct, then stores $(\mathsf{R}^*, \mathsf{r}_\mathcal{I})$ together with T^*; else calls *Claim* m_4 *Not Received* to the TTP.

The *Ticket Verification* protocol is a fair exchange protocol with the existence of an offline TTP [11] between the user and the provider of the service (a valid e-ticket is given in exchange for the permission to use the service). This enables dispute resolution protocols in case of incorrect behaviour of the actors so as to preserve the security of the system. In case of dispute, they can contact the TTP following these protocols:

Claim m_2 Not Received. This protocol can be executed if $\mathcal{U}$ sends m_1 and says that she has not received $\mathsf{m}_2 = \mathsf{V_{succ}}^*$ from $\mathcal{P}$.

Claim User $\mathcal{U}$ executes:

1. sends the ticket $\mathsf{m}_1 = \mathsf{T}^*$ to the TTP $\mathcal{T}$;

Response TTP $\mathcal{T}$ follows the next steps:

1. checks the information;
2. if the verification is correct, generates $(\mathsf{T.Sn}, \tau_3)$; then
3. signs the information $\mathsf{m}_5 = ((\mathsf{T.Sn}, \tau_3), \mathsf{sk}_\mathcal{T}(hash((\mathsf{T.Sn}, \tau_3))))$; and
4. sends m_5 to both $\mathcal{U}$ and $\mathcal{P}$. This entails acceptance of $\mathcal{U}$'s sent information and then $\mathcal{P}$ has the responsibility to unblock and send a correct m_2 to continue with the verification phase at sub-phase *verifyTicket*. After that, if the service could not be finally guaranteed, $\mathcal{U}$ could demonstrate to a third party (by showing m_5) that $\mathcal{U}$ behaved correctly and $\mathcal{P}$ was the responsible of the denial of service;

verifyTicketWithTTP Service provider $\mathcal{P}$ executes:

1. executes *verifyTicket* normally;
2. sends m_2 to both $\mathcal{T}$ and $\mathcal{U}$, and continues the *Ticket Verification* steps at point *showProof*;

Claim m_3 Not Received. This protocol can be executed if $\mathcal{P}$ sends m_2 and says that has not received $\mathsf{m}_3 = \mathsf{A}_\mathcal{U}$ (with a correct $\mathsf{r}_\mathcal{U}$ inside) from $\mathcal{U}$.

Claim Provider $\mathcal{P}$ executes:

1. blocks the ticket $\mathsf{T.Sn}$ till the reception of m_3 from $\mathcal{U}$ or m_5 by $\mathcal{T}$;
2. another $\mathsf{T.Sn}'$ received from the same connection could not be accepted and $\mathsf{m}_2 = \mathsf{V_{succ}}^*$ would be repeatedly sent in order to unblock the ticket identified by $\mathsf{T.Sn}$.

Claim m_4 Not Received. This protocol can be executed if $\mathcal{U}$ sends m_3 and says that has not received $\mathsf{m}_4 = \mathsf{R}^*$ (with the contained $\mathsf{r}_\mathcal{I}$) from $\mathcal{P}$.

Claim User $\mathcal{U}$ follows the next steps:

1. sends to the TTP $\mathcal{T}$: $(\mathsf{m}_1, \mathsf{m}_2, \mathsf{m}_3) = (\mathsf{T}^*, \mathsf{V_{succ}}^*, (\mathsf{T.Sn}, \mathsf{A}_\mathcal{U}))$;

Response TTP $\mathcal{T}$ executes:

1. checks the entire information; if verification fails, aborts the claim;
2. computes $\mathsf{A}_\mathcal{P} = PRNG(\mathsf{h_K}) \oplus \mathsf{r}_\mathcal{I}$ using K and $\mathsf{r}_\mathcal{I}$. Note that K and $\mathsf{r}_\mathcal{I}$ can be obtained by decrypting $\delta_{\mathcal{T},\mathcal{P}}$;
3. checks $\mathsf{A}_\mathcal{U}$ obtaining a valid $\mathsf{r}_\mathcal{U}$ that matches with $\mathsf{T.h}_{\mathsf{r}_\mathcal{U}}$;
4. checks $\mathsf{A}_\mathcal{P}$ obtaining a valid $\mathsf{r}_\mathcal{I}$ that matches with $\mathsf{T.h}_{\mathsf{r}_\mathcal{I}}$;
5. if everything is successful, then generates $(\mathsf{T.Sn}, \mathsf{A}_\mathcal{P}, \mathsf{A}_\mathcal{U}, \tau_4)$; otherwise, publishes which entity behaved corruptly;
6. signs the information $\mathsf{m}_6 = ((\mathsf{T.Sn}, \mathsf{A}_\mathcal{P}, \mathsf{A}_\mathcal{U}, \tau_4), \mathsf{sk}_\mathcal{T}(hash((\mathsf{T.Sn}, \mathsf{A}_\mathcal{P}, \mathsf{A}_\mathcal{U}, \tau_4)))$;
7. sends m_6 to $\mathcal{U}$.

3.2 Multiple Providers

The described proposal considers that only one provider is able to give a certain service, enabling then offline verification. Nevertheless, this scenario could be extended to the existence of multiple providers that give a certain service and accept the same ticket in different places, but guaranteeing the control of ticket overspending through online verification between all the providers. In this extended scenario, $\mathsf{sk}_\mathcal{P}$ would be shared between the providers, enabling the access to K and $\mathsf{r}_\mathcal{I}$. There would be also special care to the distribution and control of used tickets (controlled by the existence of $\mathsf{r}_\mathcal{U}$ in the database for that ticket). There should be a central database where all the providers could store all the used tickets, and then the verification would be online by imperative. Expired tickets could be removed from the database for storage efficiency, and, moreover, only ticket serial number could be stored in the database despite storing all the ticket information.

4 Security and Privacy of the System

Proposition 1. *The proposed e-ticketing system preserves authenticity, non-repudiation, integrity and the expiry date of the e-ticket.*

Claim (1). It is computationally unfeasible to make a new fraudulent e-ticket.

Security Argument. A valid e-ticket has the form $\mathsf{T}^* = (\mathsf{T}, \mathsf{Sign}_{\mathcal{I}}(\mathsf{T}))$. Then, the first step that the provider $\mathcal{P}$ does when an e-ticket is received is the verification of the signature. The *Ticket Verification* protocol will continue only if this verification ends correctly; otherwise, $\mathcal{P}$ refuses $\mathcal{U}$'s request. Thus, making a new fraudulent valid e-ticket would be equivalent to break the signature scheme and that would be computationally unfeasible as we have supposed that the issuer $\mathcal{I}$ uses a secure signature scheme.

Claim (2). The issuer can not deny the emission of a valid e-ticket.

Security Argument. A valid e-ticket has $\mathcal{I}$'s signature and the signature scheme used is secure. Consequently, the identity of the issuer is associated to the ticket; the signature is a non-repudiation evidence of origin.

Claim (3). The content of the e-ticket can not be modified.

Security Argument. Suppose that someone modifies the content of the ticket, then a new $\mathcal{I}$'s signature has to be generated over the modified content; otherwise, the e-ticket will not be valid. Again, if it is computationally unfeasible to forge the $\mathcal{I}$'s signature, it is unfeasible to modify the content of the e-ticket.

Claim (4). The e-ticket will not be longer valid after the ticket validity time $\mathsf{T.Tv}$.

Security Argument. The provider $\mathcal{P}$ receives the e-ticket from the user at the *Ticket Verification* protocol before allowing access to the service. First of all $\mathcal{P}$ checks the correctness of the e-ticket (obviously that includes the verification of $\mathsf{T.Tv}$). If the verification is not correct $\mathcal{P}$ stops the protocol and the user will have no access to the service. Also, according the *Claim 3*, the user can not tamper $\mathsf{T.Tv}$.

Result 1. *According to the definitions given at Section 3 and the Claims 1, 2, 3 and 4 we can assure that the protocol achieves the properties specified at Proposition 1.*

Proposition 2. *The e-ticketing system described in Section 3 is anonymous. The service offered is revocable anonymous.*

Claim (5). An e-ticket is anonymous.

Security Argument. A valid e-ticket has the following information $\mathsf{T} = (\mathsf{Sn}, \mathsf{Sv}, \mathsf{Pseu}_{\mathcal{U}}, \mathsf{Tv}, \mathsf{Ti}, \mathsf{h}_{\mathsf{r}_{\mathcal{I}}}, \mathsf{h}_{\mathsf{r}_{\mathcal{U}}}, \delta_{\mathcal{T},\mathcal{P}}, \ldots)$. The information related to the user's identity is solely $\mathsf{Pseu}_{\mathcal{U}} = (\mathsf{y}_{\mathcal{U}}, \mathsf{sk}_{\mathcal{I}}(hash(\mathsf{y}_{\mathcal{U}})))$, where $\mathsf{y}_{\mathcal{U}} = \alpha^{\mathsf{x}_{\mathcal{U}}} \pmod p$. The user's identity is $\mathsf{x}_{\mathcal{U}}$, thus an enemy has to solve the problem of computing the discrete logarithm to know the identity of the user. Currently no efficient algorithms are known to compute that.

Claim (6). The purchase of an e-ticket is anonymous.

Security Argument. As the protocol in Section 3.1 specifies, the channel between $\mathcal{U}$ and $\mathcal{I}$ of the ticket is anonymous. The protocol uses a Schnorr's ZKP to provide the user identity to the $\mathcal{I}$, so that the issuer can be sure that the connected user who wants to buy the ticket is the right holder of the pseudonym $\mathsf{Pseu}_{\mathcal{U}}$ without disclosing her real identity. Thus, the user does not need to reveal her identity to buy an e-ticket.

Claim (7). A fake user cannot buy an e-ticket impersonating other user.

Security Argument. In order to buy a ticket, the user has to perform a Schnorr's ZKP to prove knowledge of the identity to the issuer without revealing it. The user has to compute w_1 such as $\alpha^{\mathsf{w}_1} \stackrel{?}{=} A_1 \cdot \mathsf{y}_{\mathcal{U}}{}^{\mathsf{c}} \pmod{p}$. As far as any user preserves the privacy of her identity $\mathsf{x}_{\mathcal{U}}$ (which links to $\mathsf{Cert}_{\mathcal{U}}$ through the cooperation of $\mathcal{T}$), anyone else will not be able to compute such w_1. In this case, user can only be accused through $\mathsf{x}_{\mathcal{U}}$ of ticket overspending (supposing that the user keeps $\mathsf{x}_{\mathcal{U}}$ secretly), because she solely has the information to perform the *Ticket Verification* protocol. Thus, the e-ticketing system also preserves the exculpability property.

Result 2. *According to the definitions given at Section 3 and the Claims 5, 6 and 7 we can assert the Proposition 2. The e-ticketing system is anonymous and this anonymity could be revocable in case of a user's fraudulent action. The pseudonym manager $\mathcal{T}$ knows the correspondence between $\mathsf{x}_{\mathcal{U}}$ and $\mathsf{y}_{\mathcal{U}}$ (see Algorithm: 'Pseudonym Renewal'). Therefore, $\mathcal{T}$ could reveal the association between $\mathsf{x}_{\mathcal{U}}$ and $\mathsf{y}_{\mathcal{U}}$ due to law enforcement (e.g. a judge could request the user's identity to $\mathcal{T}$).*

Proposition 3. *The protocol satisfies the property of exculpability.*

Claim (8). The service provider can not falsely accuse the user of ticket overspending.

Security Argument. When the service provider $\mathcal{P}$ receives the message m_1 in step 1 of the *Ticket Verification* protocol *(showTicket)*, $\mathcal{P}$ looks for the ticket that matches with the received serial number in its database. If the ticket had been already spent, the service provider will find the overspending proof $\mathsf{r}_{\mathcal{U}}$ together with the ticket. The service provider has to show this element to accuse the user of overspending. If the user had not validated the ticket before, then the service provider does not have the element ($\mathcal{U}$ will send it in step 3: *showProof*), as the *hash* invertible function is believed to be computationally infeasible, and also there can not exist collisions in this *hash* function, so $\mathcal{P}$ can not falsely accuse the user of overspending. If the service provider, even not being able to prove the overspending, decides to deny the service to the user, the user can contact the TTP in order to solve the situation through *Claim* m_2 *Not Received*.

Claim (9). The user $\mathcal{U}$ is able to prove that she has already validated the ticket.

Security Argument. If a user $\mathcal{U}$ executes successfully the *Ticket Verification* protocol, $\mathcal{U}$ will obtain the exculpability proof $\mathsf{r}_{\mathcal{I}}$. She can use this proof to

demonstrate that the ticket has been validated. If the *Ticket Verification* protocol is stopped and $\mathcal{U}$ does not obtain the exculpability proof after the revelation of $\mathsf{r}_{\mathcal{U}}$, she can execute *Claim* m_4 *Not Received*. This way $\mathcal{U}$ would obtain an alternative exculpability proof from the TTP.

Result 3. *According to the definitions given at Section 3 and the Claims 8 and 9 we can assure that the protocol achieves the property specified at the Proposition 3. The ticket verification process is a fair exchange: any part can obtain the exculpability proof of the other part without revealing its own proof.*

Proposition 4. *The protocol avoids overspending with minimum requirements of persistent connections with a centralized system.*

Claim (10). The protocol avoids overspending.

Security Argument. If a user tries to overspend a ticket, then initiates the verification another time after the first validation of the ticket. The user sends T^* to $\mathcal{P}$ *(showTicket)* and waits for a response. $\mathcal{P}$ executes *(verifyTicket)* and looks for the ticket T^* in its database. Overspending can be prevented; the provider can prove it while $\mathcal{P}$ can not accuse the user of overspending attempts if $\mathcal{P}$ can not provide $\mathsf{r}_{\mathcal{U}}$.

Claim (11). If the ticket can only be validated by one provider then the verification is totally offline.

Security Argument. The provider $\mathcal{P}$ maintains a database with the serial numbers of the e-tickets that have been already validated (together with their exculpability proofs) until their expiry date. With the contents of this database the provider has enough information to decide if $\mathcal{P}$ accepts and validates a new ticket because $\mathcal{P}$ can check both the issuer's signature and the fact that the e-ticket has not been spent before. So the provider does not need to contact to any party during the validation of an e-ticket.

Claim (12). If the ticket can be validated with several providers then the providers must be connected and share a database of spent tickets.

Security Argument. The set of providers maintains a shared database with the serial numbers of the e-tickets that have been already validated (together with their exculpability proofs) until their expiry date. The contents of this database are used by the providers to decide if they accept and validate a new ticket. So the provider does not need connection to the issuer during the verification of an e-ticket, but the set of providers must have a shared database instead.

Result 4. *According to the definitions given at Section 3 and the Claims 10, 11 and 12 we can assure that the protocol achieves Proposition 4. The issuer is offline during the verification phase and the providers contact among them only in some kind of services. In all cases, the protocol prevents ticket overspending.*

5 Conclusions

We have presented an e-ticketing scheme with revocable anonymity, and exculpability as a novel security requirement. Moreover, we have considered the use of personal mobile devices in the system, trying to lighten then the computational requirements on the users' side. A new scenario has been introduced since we assume, for simplicity, that only one provider is able to give a certain service; then, the ticket verification phase is totally offline from the ticket issuer. But, if multiple providers were able to give that service, then ticket verification should be actively centralized (online verification between all the service providers) in order to avoid ticket overspending attempts. An analysis of the security and privacy of the proposal has been also performed, breaking down all the security requirements and evaluating the achievements. The future work is to implement the proposal for mobile devices with contactless communication.

Disclaimer and Acknowledgements

Partial support by the Spanish MICINN(projects TSI2007-65406-C03-01, ARES-CONSOLIDER INGENIO 2010 CSD2007-00004, TSI2007-62986, "RIPUP" TIN2009-11689), the Spanish Ministry of Industry, Commerce and Tourism (project eVerification TSI-020100-2009-720), and the Government of Catalonia (grant 2009 SGR1135) is acknowledged. The authors are solely responsible for the views expressed in this paper, which do not necessarily reflect the position of UNESCO nor commit that organization.

References

1. Bao, F.: A scheme of digital ticket for personal trusted device. In: 15th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC 2004), vol. 4, pp. 3065–3069. IEEE, Los Alamitos (2004)
2. Chaum, D.: Blind signatures for untraceable payments. In: Advances in Cryptology - CRYPTO 1982, pp. 199–203 (1983)
3. Chen, Y.-Y., Chen, C.-L., Jan, J.-K.: A mobile ticket system based on personal trusted device. Wireless Personal Communications: An International Journal 40(4), 569–578 (2007)
4. Dingledine, R., Mathewson, N., Syverson, P.: Tor: The second-generation onion router. In: Proceedings of the 13th USENIX Security Symposium (2004)
5. Elliot, J.: The one-card trick multi-application smart card e-commerce prototypes. Computing & Control Engineering Journal 10(3), 121–128 (1999); IET
6. Fujimura, K., Kuno, H., Terada, M., Matsuyama, K., Mizuno, Y., Sekine, J.: Digital-ticket-controlled digital ticket circulation. In: 8th USENIX Security Symposium, pp. 229–240. USENIX (1999)
7. Haneberg, D.: Electronic ticketing: risks in e-commerce applications. In: Digital Excellence, pp. 55–66. Springer, Heidelberg (2008) ISBN 3540726209
8. Haneberg, D., Stenzel, K., Reif, W.: Electronic-onboard-ticketing: Software challenges of an state-of-the-art m-commerce application. In: Pousttchi, K., Turowski, K. (eds.) Workshop Mobile Commerce. Lecture Notes in Informatics (LNI), vol. 42, pp. 103–113. Gesellschaft für Informatik, GI (2004)

9. Heydt-Benjamin, T.S., Chae, H.-J., Defend, B., Fu, K.: Privacy for public transportation. In: Danezis, G., Golle, P. (eds.) PET 2006. LNCS, vol. 4258, pp. 1–19. Springer, Heidelberg (2006)
10. Gans, G.K., Hoepman, J.-H., Garcia, F.D.: A practical attack on the mifare classic. In: Grimaud, G., Standaert, F.-X. (eds.) CARDIS 2008. LNCS, vol. 5189, pp. 267–282. Springer, Heidelberg (2008)
11. Kremer, S., Markowitch, O., Zhou, J.: An intensive survey of fair non-repudiation protocols. Computer Communications 25, 1606–1621 (2002)
12. Kuramitsu, K., Sakamura, K.: Electronic tickets on contactless smartcard database. In: Hameurlain, A., Cicchetti, R., Traunmüller, R. (eds.) DEXA 2002. LNCS, vol. 2453, pp. 392–402. Springer, Heidelberg (2002)
13. Kuramitsu, K., Murakami, T., Matsuda, H., Sakamura, K.: Ttp: Secure acid transfer protocol for electronic ticket between personal tamper-proof devices. In: 24th Annual International Computer Software and Applications Conference (COMPSAC 2000), Taipei, Taiwan, vol. 24, pp. 87–92 (October 2000)
14. Matsuo, S., Ogata, W.: Electronic ticket scheme for its. IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences E86A(1), 142–150 (2003)
15. Patel, B., Crowcroft, J.: Ticket based service access for the mobile user. In: Proceedings of the 3rd Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM 1997), Budapest, Hungary, pp. 223–233 (1997)
16. Quercia, D., Hailes, S.: Motet: Mobile transactions using electronic tickets. In: Proceedings 1st International Conference on Security and Privacy for Emerging Areas in Communications Networks, Athens, Greece, vol. 24, pp. 374–383 (September 2005)
17. Schnorr, C.-P.: Efficient signature generation by smart cards. J. Cryptology 4(3), 161–174 (1991)
18. Siu, W.I., Guo, Z.S.: Application of electronic ticket to online trading with smart card technology. In: Proceedings of the 6th INFORMS Conference on Information Systems and Technology (CIST 2001), Florida (US), pp. 222–239 (2001)
19. Siu, W.I., Guo, Z.S.: The secure communication protocol for electronic ticket management system. In: 8th Asia-Pacific Software Engineering Conference (APSEC 2001), University of Macau (2001)

Privacy Enforcement and Analysis for Functional Active Objects

Florian Kammüller

Middlesex University, London and Technische Universität Berlin
f.kammueller@mdx.ac.uk, flokam@cs.tu-berlin.de

Abstract. In this paper we present an approach for the enforcement of privacy in distributed active object systems, illustrate its implementation in the language ASP_{fun}, and formally prove privacy based on information flow security.

1 Introduction

The language ASP_{fun} is a calculus of active objects. It has been developed in the interactive theorem prover Isabelle/HOL – in a fully formal co-development style [HK09]. ASP_{fun} is a functional language using the Theory of Objects by Abadi and Cardelli for local object calculation but adding a second layer of distributed *activities* that communicate asynchronously via *futures*.

Activities are a unification of objects and processes having – like objects – a local data space while representing a unit of distribution containing a queue of currently processed calls to methods of this activity's object.

Futures are promises to the results of method calls. They realize asynchronous communication because an activity that calls a method in another activity immediately receives a future as a result. Thus, the calling activity can continue its current calculation. Only when the calling activity needs the actual result of the method call, a so-called *wait-by-necessity* could occur. In Proactive, an imperative Java implementation of distributed active objects for Grid-computing, this is indeed the case. In our functional computation model ASP_{fun}, however, we allow the return of possibly not fully evaluated methods calls. We completely avoid wait-by-necessity and possibly resulting dead-lock situations.

Moreover, we can fully prove in our Isabelle/HOL formalization that ASP_{fun} is non-blocking. This proof actually comes as a by-product of the proof of type safety we have conducted in Isabelle/HOL for our ASP_{fun} type system [HK09]. The suitability of ASP_{fun} as a privacy enforcement language has already been demonstrated [Kam10]: based on the semantic primitive of active object *update*, an economic and clean implementation of data hiding is possible.

The contribution of this paper is a concept more generally useful for privacy: *flexible parameterization* – enabling the use of service functions while not supplying all parameters. For example, in the European project SENSORIA the COWS calculus has been designed as an extension to the Pi-calculus to realize *correlation* a similarly dynamic service concept [BNRNP08].

J. Garcia-Alfaro et al. (Eds.): DPM 2010 and SETOP 2010, LNCS 6514, pp. 93–107, 2011.

In ASP_{fun}, we get this privacy enhancing use of services for free via the flexibility provided through *functional* replies. This idea has been tested through practical applications with a prototypical ASP_{fun} implementation in Erlang [FK10]. Now, we formalize this idea and prove its security using the well-established method of information flow analysis. More precisely, our contribution consists in the following steps.

- We provide a security model for ASP_{fun} for formal security analysis, i.e. a formal definition of noninterference for functional active objects.
- Based on the construction of a function-like currying for our object calculus, we provide a generic way of implementing flexible parameterization.
- Using the formal notion of noninterference, we prove that flexible parameterization is secure, and thus provides privacy.

As a global methodology we aim at providing a double-sided method complementing privacy enforcement with an accompanying analysis. To round off the paper, we present a concept for a modular assembly kit for security for ASP_{fun} (in the style of Mantel's successful approach) that enables a systematic representation of language based security properties for distributed active object systems, thereby modelling privacy.

2 Functional Active Objects with ASP_{fun}

The language ASP_{fun} [HK09] is a computation model for functional active objects. Its local object language is a simple ς-calculus [AC96] featuring method call $t.l(s)$, and method update $t.l := \varsigma(x,y)b$ on objects (ς is a binder for the self x and method parameter y). Objects consist of a set of labeled methods $[l_i = \varsigma(x,y)b]^{i\in 1..n}$ (attributes are considered as methods with no parameters). ASP_{fun} now simply extends this basic object language by a command *Active(t)* for creating an activity for an object t. A simple configuration containing just activities α and β within which are so-called active objects t and t' is depicted in Figure 1. This figure also illustrates *futures*, a concept enabling asynchronous communication. Futures are promises for the results of remote method calls, for example in Figure 1, f_k points to the location in activity β where at some point the result of the method evaluation $t'.l(s)$ can be retrieved from. Futures are first class citizen but they are not part of the *static* syntax of ASP_{fun}, that is, they cannot be used by a programmer. Similarly, activity references, e.g. α, β, in Figure 1, are references and not part of the static syntax. Instead, futures and activity references constitute the machinery for the computation of configurations of active objects. ASP_{fun} is built as a conceptual simplification of ASP [CH05] – both languages support the Java API Proactive [Pro08].

2.1 Informal Semantics of ASP_{fun}

Local (ς-calculus) and *parallel* (configuration) semantics are given by a set of reduction rules informally described as follows.

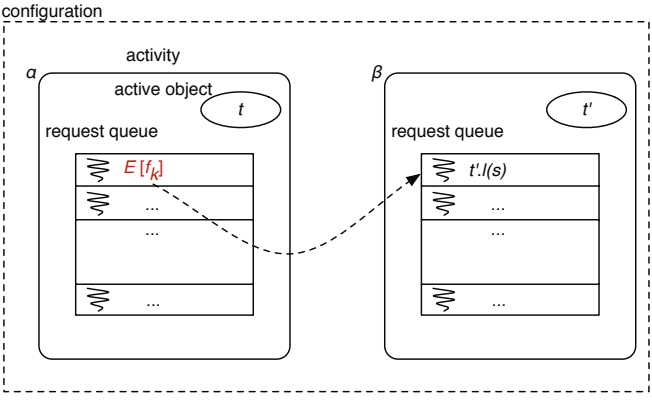


Fig. 1. ASP_{fun}: a configuration

- LOCAL: the local reduction relation $\rightarrow_\varsigma$ is based on the ς-calculus.
- ACTIVE: $Active(t)$ creates a new activity $\alpha[\varnothing, t]$ for new name α, empty request queue, and with t as active object.
- REQUEST: *method call* $\beta.l$ creates new future f_k in future-list of activity β (see Figure 1).
- REPLY: *returns result*, i.e. replaces future f_k by the referenced result term s (possibly not fully evaluated).
- UPDATE-AO: *activity update* creates a copy of the activity and updates the active object of the copy – the original remains the same (functional active objects are *immutable*).

2.2 Formal ASP_{fun} Semantics

We use a concise contextual description with contexts E defined as usual. Classically we define contexts as expressions with a single hole ($\bullet$).

$$\begin{aligned} E ::= \bullet \mid [l_i = \varsigma(x,y)E, l_j = \varsigma(x_j,y_j)t_j^{j\in(1..n)-\{i\}}] \mid E.l_i(t) \mid \\ s.l_i(E) \mid E.l_i := \varsigma(x,y)s \mid s.l_i := \varsigma(x,y)E \mid Active(E) \end{aligned}$$

$E[s]$ denotes the term obtained by replacing the single hole by s. The semantics of the ς-calculus is then given by the following two reduction rules for calling and updating a method (or field) of an object.

CALL

$$\frac{l_i \in \{l_j\}^{j\in 1..n}}{\begin{array}{l} E\left[[l_j = \varsigma(x_j,y_j)b_j]^{j\in 1..n}.l_i(b)\right] \rightarrow_\varsigma \\ E\left[b_i\{x_i \leftarrow [l_j = \varsigma(x_j,y_j)b_j]^{j\in 1..n}, y_j \leftarrow b\}\right] \end{array}}$$

UPDATE

$$\frac{l_i \in \{l_j\}^{j\in 1..n}}{\begin{array}{l} E\left[[l_j = \varsigma(x_j,y_j)b_j]^{j\in 1..n}.l_i := \varsigma(x,y)b\right] \rightarrow_\varsigma \\ E\left[[l_i = \varsigma(x,y)b, l_j = \varsigma(x_j,y_j)b_j^{j\in(1..n)-\{i\}}]\right] \end{array}}$$

Table 1. $\mathrm{ASP}_{\mathrm{fun}}$ semantics

$$\textsc{local}\quad \frac{s \to_\varsigma s'}{\alpha[f_i \mapsto s :: Q, t] :: C \to_\parallel \alpha[f_i \mapsto s' :: Q, t] :: C}$$

$$\textsc{active}\quad \frac{\gamma \notin (\mathrm{dom}(C) \cup \{\alpha\}) \qquad noFV(s)}{\alpha[f_i \mapsto E[Active(s)] :: Q, t] :: C \to_\parallel \alpha[f_i \mapsto E[\gamma] :: Q, t] :: \gamma[\varnothing, s] :: C}$$

$$\textsc{request}\quad \frac{f_k \text{ fresh} \qquad noFV(s)}{\alpha\,[f_i \mapsto E[\beta.l(s)] :: Q, t] :: \beta[R, t'] :: C \to_\parallel \alpha\,[f_i \mapsto E[f_k] :: Q, t] :: \beta\,[f_k \mapsto t'.l(s) :: R, t'] :: C}$$

$$\textsc{self-request}\quad \frac{f_k \text{ fresh} \qquad noFV(s)}{\alpha\,[f_i \mapsto E[\alpha.l(s)] :: Q, t] :: C \to_\parallel \alpha\,[f_k \mapsto t.l(s) :: f_i \mapsto E[f_k] :: Q, t] :: C}$$

$$\textsc{reply}\quad \frac{\beta[f_k \mapsto s :: R, t'] \in \alpha[f_i \mapsto E[f_k] :: Q, t] :: C}{\alpha[f_i \mapsto E[f_k] :: Q, t] :: C \to_\parallel \alpha[f_i \mapsto E[s] :: Q, t] :: C}$$

$$\textsc{update-AO}\quad \frac{\gamma \notin (\mathrm{dom}(C) \cup \{\alpha\}) \qquad noFV(\varsigma(x,y)s) \qquad \beta[R, t'] \in (\alpha[f_i \mapsto E[\beta.l := \varsigma(x,y)s] :: Q, t] :: C)}{\alpha[f_i \mapsto E[\beta.l := \varsigma(x,y)s] :: Q, t] :: C \to_\parallel \alpha[f_i \mapsto E[\gamma] :: Q, t] :: \gamma[\varnothing, t'.l := \varsigma(x,y)s] :: C}$$

The semantics of $\mathrm{ASP}_{\mathrm{fun}}$ is built over the local semantics of the ς-calculus as a reduction relation $\to_\parallel$ that we call the parallel semantics (see Table 1).

2.3 Broker Example

The following example illustrates the advantages of futures for the implementation of services. The three activities hotel, broker, and customer are composed by $\parallel$ into a configuration. The customer wants to make a hotel reservation in hotel. He uses a broker for this service by calling a method book provided in the active object of the broker. We omit the actual search of the broker in his database and instead hardwire the solution to always contact some hotel. That is, the method book is implemented as a call $\varsigma(x, date)$hotel.room($date$) to a function room in the hotel. Also the internal administration of hotel is omitted; its method room just returns a constant bookingreference bookingref. The dependence of this bookingref on the parameter *date* exists only implicitly. Therefore, we denote it by $\mathrm{bookingref}_{\langle date \rangle}$ in instantiations. Initially, only the future list of the customer contains a request for a booking to broker; the future lists of broker and hotel are empty. The following steps of the semantic reduction relation $\to_\parallel$ illustrate how iterated application of reduction rules evaluates the program.

$$\begin{array}{l} \text{customer}[f_0 \mapsto \text{broker.book}(\mathit{date}), t] \\ \parallel \text{broker}[\varnothing, [\text{book} = \varsigma(x, \mathit{date})\text{hotel.room}(\mathit{date}), \ldots]] \\ \parallel \text{hotel}[\varnothing, [\text{room} = \varsigma(x,\mathit{date})\text{bookingref}, \ldots]] \end{array}$$

The following step of the semantic reduction relation $\rightarrow_{\parallel}^{*}$ creates the new future f_1 in broker by rule REQUEST, this call is reduced according to LOCAL, and the original call in the customer replaced by f_1.

$$\begin{array}{l} \text{customer}[f_0 \mapsto f_1, t] \\ \parallel \text{broker}[f_1 \mapsto \text{hotel.room}(\mathit{date}), \ldots] \\ \parallel \text{hotel}[\varnothing, [\text{room} = \varsigma(x,\mathit{date})\text{bookingref}, \ldots]] \end{array}$$

The parameter x representing the *self* is not used but the call to hotel's method room with parameter *date* creates again by rule REQUEST a new future in the request queue of the hotel activity that is immediately reduced due to LOCAL to bookingreference where the index indicates that *date* has been used.

$$\begin{array}{l} \text{customer}[f_0 \mapsto f_1, t] \\ \parallel \text{broker}[f_1 \mapsto f_2, \ldots] \\ \parallel \text{hotel}[f_2 \mapsto \text{bookingref}_{\langle \mathit{date} \rangle}, \ldots] \end{array}$$

Finally, the result bookingreference is returned to the client by two REPLY-steps: first the future f_2 is returned from the broker to the customer and then this client receives the bookingreference via f_2 directly from the hotel.

$$\begin{array}{l} \text{customer}[f_0 \mapsto \text{bookingref}_{\langle \mathit{date} \rangle}, t] \\ \parallel \text{broker}[f_1 \mapsto f_2, \ldots] \\ \parallel \text{hotel}[f_2 \mapsto \text{bookingref}_{\langle \mathit{date} \rangle}, \ldots] \end{array}$$

This configuration can be considered as the final one; at least the service has been finished. From the perspective of privacy, it is actually here that we would like to end the evaluation. Unfortunately, the future f_2 is also available to the broker. So, in an final step the broker can serve himself the bookingreference as well.

$$\begin{array}{l} \text{customer}[f_0 \mapsto \text{bookingref}_{\langle \mathit{date} \rangle}, t] \\ \parallel \text{broker}[f_1 \mapsto \text{bookingref}_{\langle \mathit{date} \rangle}, \ldots] \\ \parallel \text{hotel}[f_2 \mapsto \text{bookingref}_{\langle \mathit{date} \rangle}, \ldots] \end{array}$$

The abstract general semantics of ASP_{fun} allows this privacy breach.

We introduce now a general way of enforcing privacy by not disclosing private data in the first place. We show that relying on the ASP_{fun} paradigm guarantees that flexible parameterization can be used to use services in a private manner.

3 Flexible Parameterization

We use the example of the hotel broker again to show how flexible parameterization may be used to keep private data in one's secure local environment. As an informal security policy we assume that the client does not want to disclose

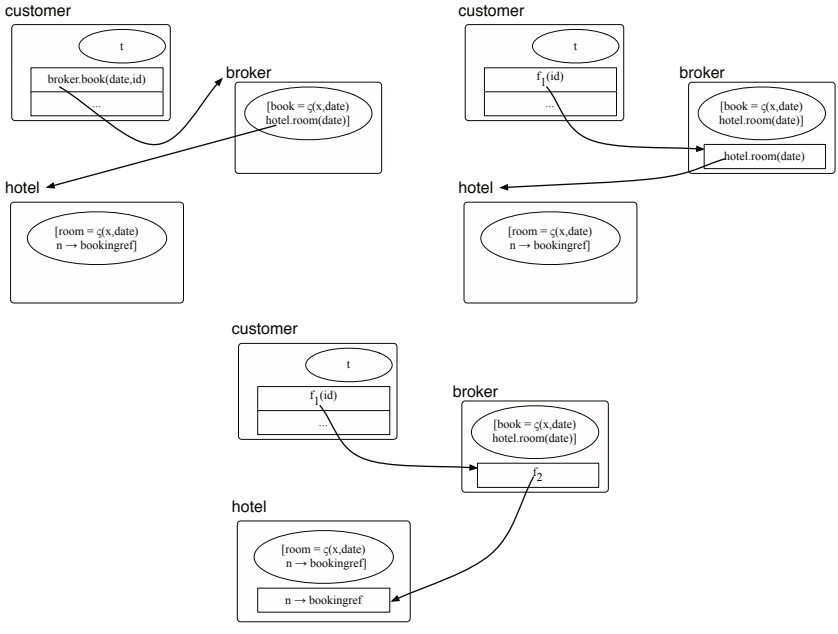


Fig. 2. Flexible parameterization: delegation to partially instantiatable room method

his name to the broker. In Figure 2, we see the same scenario as in the previous section's example but with a slightly generalized program. Here, the function room in hotel has an additional parameter id besides date. However, room can be called just supplying the first date parameter. The broker still delegates the partially instantiated request to the hotel. Thereby, the customer can then directly access a function in hotel – via the futures f_1 and f_2 – that calculates his bookingref on supplying the missing parameter id. The broker can also access the resulting function but not the private data of the customer as it does not need to be transmitted. We have implemented this technique in our Erlang prototype for ASP_{fun} [FK10] as a pragmatic extension of the base language. However, as we will show now, this feature on flexible parameterization can be constructed conservatively in ASP_{fun} using *currying*.

Currying is a well known technique in functional programming to render functions with several parameters partially applicable. That is, the parameters of a curried function may be supplied one after the other, in each step returning a new function.

Recall the definition of curry and its inverse uncurry in the λ-calculus.

$$\begin{aligned} \mathsf{curry} &\equiv \lambda\ f\ p.\ f(\mathsf{fst}\ p)(\mathsf{snd}\ p) \\ \mathsf{uncurry} &\equiv \lambda\ f\ a\ b.\ f(a,b) \end{aligned}$$

Here, (a, b) denotes the product and fst and snd the corresponding projections on a and b respectively. This datatype is itself definable in terms of simpler λ-terms as follows.

$$\begin{aligned}(a, b) &\equiv \lambda\ f.\ f\ a\ b\\ \mathsf{fst}\ p &\equiv (\lambda\ x\ y.\ x)\\ \mathsf{snd}\ p &\equiv (\lambda\ x\ y.\ y)\end{aligned}$$

We recall these classic definitions in order to prepare the less intuitive definition of currying for the ς-calculus and hence for $\mathrm{ASP}_{\mathrm{fun}}$.

3.1 Currying in $\mathbf{ASP_{fun}}$

In comparison to the ς-calculus, the base objects of $\mathrm{ASP}_{\mathrm{fun}}$ differ in that we explicitly introduce a second parameter to each method in addition to the *self*-parameter x. Therefore, when we emulate functions in our version of the ς-calculus we profit from this parameter and avoid roundabout ways of encoding parameters.[1] As a prerequisite for having several parameters, we need products. Compared to the above presented encoding of pairs in the λ-calculus, pairs in the ς-calculus can make use of the natural *record* structure of objects thus rendering a more intuitive notion of product as follows.

$$\begin{aligned}(a, b) &\equiv [\,\mathsf{fst}\ = \varsigma(x, y)a,\ \mathsf{snd}\ = \varsigma(x, y)b\,]\\ \mathsf{fst}\ p &\equiv p.\mathsf{fst}\\ \mathsf{snd}\ p &\equiv p.\mathsf{snd}\end{aligned}$$

We simulate currying of a method f of an object o that expects a pair p of type $\alpha \times \beta$ as second parameter, i.e.

$$o = [\,f = \varsigma(x, p).t\,]$$

by extending this object o with a second method f_C as follows.

$$\begin{aligned}\mathsf{curry}\ o \equiv [\,&f = \varsigma(x, p)o.f(p),\\ &f_C = \varsigma(x, a)[f' = \varsigma(y, b)x.f(a, b)]\,]\end{aligned}$$

3.2 Hiding

We introduce formal definitions of hiding and restrictions that will be used later for the formal definition of strong noninterference. Hiding, at the object level, is an operation that takes a ς-object and a label and hides the methods that are identified by the label. Practically, it is realized by overwriting the field labeled l of a ς-object with the empty object $\langle\rangle$.

[1] In the ς-calculus the parameter has to be simulated by updating a separate field in an objects and that consequently needs to be attached to each object.

Definition 1 (Hiding for ς-terms)

$$(o.m)(p) \setminus l \equiv \text{ if } m = l \text{ then } \langle\rangle \text{ else } (o \setminus l).m(p \setminus l)$$

$$[l_j = \varsigma(x_j, y_j)b_j]^{j\in 1..n} \setminus l \equiv \begin{cases} [l_i = \varsigma(x,y)\langle\rangle, l_j = \varsigma(x_j,y_j)b_j]^{j\in 1..n-\{i\}}, & \text{if } l_i = l \\ [l_j = \varsigma(x_j,y_j)(b_j \setminus l)]^{j\in 1..n}, & \text{else} \end{cases}$$

$$(o.m := f) \setminus l \equiv \text{ if } m = l \text{ then } o.m := \langle\rangle \text{ else } (o \setminus l).m := f$$

To lift the hiding operator to configurations, we just have to introduce name spacing with respect to configuration names and map that to the previous hiding definition.

Definition 2 (Hiding for ASP_{fun} terms). *For any configuration C hiding is defined as follows.*

$$(\alpha[Q,t] :: C) \setminus \Lambda.l \equiv \begin{cases} \alpha[Q,t] :: (C \setminus \Lambda.l) & \text{if } \alpha \neq \Lambda \\ \alpha[Q, t \setminus l] :: (C \setminus \Lambda.l) & \text{else} \end{cases}$$

In order to define a notion of security for ASP_{fun}, we define observational equivalence. This means informally that programs are considered to be secure if an attacker cannot tell apart the outcomes of the program when observing only those parts that are visible to him according to a security assignment. For simplicity, we assume this security assignment for ASP_{fun} programs to be $S : C \to \{\Delta, \nabla\}$. That is, we assume that all activities of a configuration C can be divided into only two groups: the secure part represented by Δ (usually called high H) and the insecure part represented by Δ's complement $\nabla \equiv \text{dom}(C) - \Delta$ (or low L). [2] In general, this is not such an unrealistic assumption if one takes the viewpoint of the secure part: the rest of the world is either part of the secure domain or not.

Definition 3 (Security for ASP_{fun}). *Two configurations C_0, C_1 are equivalent for the attacker $C_0 =_\nabla C_1$ if their visible parts in ∇ are equal modulo renaming and local reduction with $\to_\varsigma$. A configuration C_0 is now called secure, if for any other configuration C_1 that is equivalent $C_0 =_\nabla C_1$, if $C_0 \to_\parallel^* C_0'$ and $C_1 \to_\parallel^* C_1'$ such that C_0', C_1' are values, then $C_0' =_\nabla C_1'$.*

Equality up to renaming and local reduction needs a word of explanation. We assume two configurations to be equal from an attackers perspective if the "low" parts in ∇ are isomorphic: we cannot assume the names to be equal as they are generated in each run. However, given a bijection on these names, we can identify any two configurations in two runs of an ASP_{fun} program if their low parts are isomorphic. We further normalize the low equality relation modulo local reduction as two configurations should not be distinguishable if they differ only in the local evaluation of request queue entries and are otherwise isomorphic. Overloading syntax, we simply write $t \setminus \Delta$ to denote hiding the list of labels of

[2] We use here $-$ instead of $\setminus$ for symmetric set difference because $\setminus$ is now used for hiding.

all activities that are in Δ, i.e. that are considered to be high. Hiding is not a usual term operation: it takes labels as second argument, which are terms. Hence, the confluence result for terms, or the context rules [HK09], respectively, do not apply. That is, from $t \rightarrow_{\parallel} t'$ does not follow $t \setminus \Delta \rightarrow_{\parallel} t' \setminus \Delta$. In fact, this is the case if and only iff hiding is secure as we will see in the following section.

4 Noninterference via Hiding

4.1 Language Based Modular Assembly Kit for Security (LB-MAKS)

In this section we want to introduce a conceptual framework – MAKS [Man00] – enabling the comparison and proof of security properties. MAKS is based on labeled transition systems, a model too abstract for the consideration of certain security critical situations — like blockings.

MAKS is based on a labeled transition system model for systems. It defines a set of six *basic security predicates* defined as closures over set properties [Man02]. *Backwards strong deletion* (BSD) and *backwards strong insertion* (BSI) are two basic security predicates. The strongest security property in MAKS is given as the conjunction of *backwards strong deletion* and *backwards strong insertion* for the flow policy $\mathcal{V}_L^{LH}$ defined as $(L, \varnothing, H)$ – the classical policy: "information may flow from L to H but not vice versa". This "root" property is not named but it forms the root of the tree formed by implications between the identified properties. On one branch from this root lie *separability*, *nondeducibility on inputs*, *perfect security property*, *noninference* (note this is *not* noninterference) and on the other branch lie *forward correctability* and *generalized noninterference*. Both branches unite in the same property *generalized noninference* (without "ter"). The positions in this "implication" tree are implicitly proved since all these properties are expressed as conjunctions of basic security predicates.

Focardi and Gorrieri compare security properties based on trace semantics but already consider algebraic characterizations using their process algebra SPA [FG95]. Like us, motivated by the fact that bisimulation based approaches are more fine grained, they provide algebraic definitions of security properties. For example, the simple characterization of *strong nondeterministic noninterference* (SNNI) in [FG95] is as follows, where $|_\Delta$ is restriction and $\setminus\Delta$ hiding.

$$E \mid_\Delta \cong E \setminus \Delta \tag{1}$$

We adapt the original syntax to avoid confusion. This relation $\cong$ can either represent a trace set equality or – for a more refined view – a bisimulation relation. This notion of strong nondeterministic noninterference is very concise but also a strong predicate. It corresponds to $BSD_{\mathcal{V}_L^{LH}}(Tr) \wedge BSI_{\mathcal{V}_L^{LH}}(Tr)$, the root of the MAKS tree [Man00]. The open question is: how does our language based characterization of security relate to the classical notions given by Mantel, Focardi and Gorrieri, and McLean? In the following, we show that we can enforce noninterference based on hiding in our language based model.

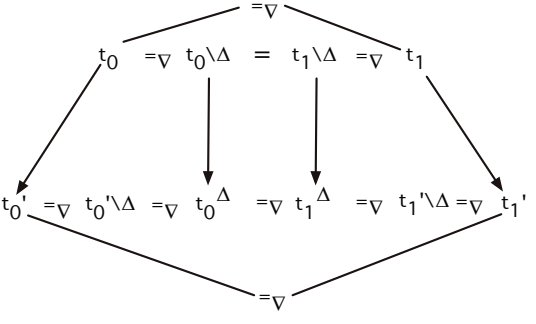
Hiding (see Section 3.2) can be used to enforce information flow control. In [Kam10], we showed that in some special cases hiding implies security. These cases use that sometimes hiding has no effect, i.e. $t \rightarrow_{\|}^{*} t'$ and $t \setminus \Delta \rightarrow_{\|}^{*} t'$.

4.2 Language Based Security Characterization

The characterization of security by hiding we have given in [Kam10] somehow implicitly assumes that $t' = t \setminus \Delta$ This assumption expresses that in secure programs the hiding operation is compositional, i.e. $t \rightarrow_{\|} t' \Rightarrow t' \setminus \Delta = t \setminus \Delta$. In an attempt to transfer the hiding based noninterference by Focardi and Gorrieri to ASP_{fun}, we derive the following theorem.

Theorem 1 (NI by Hiding). *Let t be an ASP_{fun} term representing a program and Δ be a security policy. If $t \rightarrow_{\|}^{*} t'$ implies $t \setminus \Delta \rightarrow_{\|}^{*} t^{\Delta}$ with $t^{\Delta} =_{\nabla} t' \setminus \Delta$, i.e. $\setminus$ is (low)-compositional for Δ, then t is secure in the sense of Definition 3.*

Proof. By definition of $=_{\nabla}$, $t =_{\nabla} t \setminus \Delta$ for any term t. Let $t_0 =_{\nabla} t_1$, $t_0 \rightarrow_{\|}^{*} t_0'$ and $t_1 \rightarrow_{\|}^{*} t_1'$ such that t_0', t_1' are values according to the hypotheses in the Security Definition 3.



Since $t_0 =_{\nabla} t_1$, by definition, $t_0 \setminus \Delta = t_1 \setminus \Delta$ up to isomorphism due to renaming. Since $t_0 \rightarrow_{\|}^{*} t_0'$ and $t_1 \rightarrow_{\|}^{*} t_1'$, we can apply the assumption to obtain $t_0 \setminus \Delta \rightarrow_{\|}^{*} t_0^{\Delta}$, $t_0^{\Delta} =_{\nabla} t_0' \setminus \Delta$ and $t_1 \setminus \Delta \rightarrow_{\|}^{*} t_1^{\Delta}$ and $t_1^{\Delta} =_{\nabla} t_1' \setminus \Delta$. Now, $t_0 \setminus \Delta = t_1 \setminus \Delta$ implies $t_0^{\Delta} =_{\rightarrow_{\|}^{*}} t_1^{\Delta}$ because of confluence of ASP_{fun}which gives $t_0^{\Delta} =_{\nabla} t_1^{\Delta}$. Since $t_0^{\Delta} =_{\nabla} t_0' \setminus \Delta$ and $t_1^{\Delta} =_{\nabla} t_1' \setminus \Delta$ we get in turn by transitivity of $=_{\nabla}$ that $t_0' \setminus \Delta =_{\nabla} t_1' \setminus \Delta$ under the same renaming as before (or possibly a conservative extension due to creation of new low elements). This corresponds to $t_0' =_{\nabla} t_1'$ by definition of $=_{\nabla}$ and we are finished. □

Theorem 1 proves noninterference according to Definition 3. This is a less restrictive security as the notion of strong nondeterministic noninterference, as used by Focardi and Gorrieri [FG95]. But the criteria characterizing security in Theorem 1 corresponds to the original definition (1) [FG95]. What is more, ours is a real language based security notion for real active objects in ASP_{fun} and not just abstract event systems. Conceptually, the correspondence provides an important link between the elegant world of event systems and the more tedious

but fairer language based models. It marks the first important stepping stone for LB-MAKS, our envisaged security tool kit for ASP_{fun}. In the following section, we will evaluate our notion on the running example of the hotel broker.

4.3 Application Example

In this section, we demonstrate how the security characterization given in Theorem 1 can be practically useful to statically verify security. We reconsider the hotel broker example from Section 2.3 to prove that it is insecure; we then show using the same method that the improved version using flexible parameterization by currying (Section 3) is secure. To model private data, we refine the initial configuration by introducing some private information. There is a data object containing the customer's identity "name" that he wants to keep private from the broker.

$$t \equiv \begin{array}{l} \text{data}[\varnothing, [\text{name} = \text{id}]] \\ \parallel \text{customer}[f_0 \mapsto \text{broker.book}(\text{data.name}), t] \\ \parallel \text{broker}[\varnothing, [\text{book} = \varsigma(x,(d,n))\text{hotel.room}(d,n), \ldots]] \\ \parallel \text{hotel}[\varnothing, [\text{room} = \varsigma(x,(d,n))\text{bookingref}, \ldots]] \end{array}$$

In several evaluation steps this program reduces to the following. Other than the insertion of the identity, the evaluation is little changed with respect to the earlier version in Section 2.3, apart from the fact that bookingref now also depends on the customer's id.

$$t' \equiv \begin{array}{l} \text{data}[f_1 \mapsto \text{id}, [\text{name} = \text{id}]] \\ \parallel \text{customer}[f_0 \mapsto \text{bookingref}_{\langle d,\text{id}\rangle}, [\ldots]] \\ \parallel \text{broker}[f_2 \mapsto \text{bookingref}_{\langle d,\text{id}\rangle}, [\ldots]] \\ \parallel \text{hotel}[f_3 \mapsto \text{bookingref}_{\langle d,\text{id}\rangle}, [\ldots]] \end{array}$$

We want to prove security of enforcement via hiding for the example configuration given the security policy $\Delta = \{\text{customer}, \text{data}\}, \nabla = \{\text{hotel}, \text{broker}\}$. According to Theorem 1, we need to show compositionality of hiding with respect to private data labeled data.name since this is (for simplicity) the only data in Δ.

Applying hiding of customer's identity to the initial configuration t, i.e. $t \setminus \Delta$, erases the customer's name in data.

$$\left(\begin{array}{l} \text{data}[\varnothing, [\text{name} = \text{id}]] \\ \parallel \text{customer}[f_0 \mapsto \text{broker.book}(\text{data.name}), t] \\ \parallel \text{broker}[\varnothing, [\text{book} = \varsigma(x,(d,n))\text{hotel.room}(d,n), \ldots]] \\ \parallel \text{hotel}[\varnothing, [\text{room} = \varsigma(x,(d,n))\text{bookingref}, \ldots]] \end{array}\right) \setminus \text{data.name} =$$

$$\begin{array}{l} \text{data}[\varnothing, [\text{name} = \langle\rangle]] \\ \parallel \text{customer}[f_0 \mapsto \text{broker.book}(\text{data.name}), t] \\ \parallel \text{broker}[\varnothing, [\text{book} = \varsigma(x,(d,n))\text{hotel.room}(d,n), \ldots]] \\ \parallel \text{hotel}[\varnothing, [\text{room} = \varsigma(x,(d,n))\text{bookingref}, \ldots]] \end{array}$$

Finally, this $t \setminus \Delta$ reduces to the following t^{Δ}.

$$t^{\Delta} \equiv \begin{array}{l} \quad \mathrm{data}[f_1 \mapsto \mathrm{id}, [\mathrm{name} = \langle\rangle]] \\ \| \ \mathrm{customer}[f_0 \mapsto \mathrm{bookingref}_{\langle d,\langle\rangle\rangle}, [\ldots]] \\ \| \ \mathrm{broker}[f_2 \mapsto \mathrm{bookingref}_{\langle d,\langle\rangle\rangle}, [\ldots]] \\ \| \ \mathrm{hotel}[f_3 \mapsto \mathrm{bookingref}_{\langle d,\langle\rangle\rangle}, [\ldots]] \end{array}$$

If we apply hiding "data.name" to the *reduced* configuration t' instead, we still erase the identity of the customer in data but do not erase the id that is now implicit in bookingref.

$$\left(\begin{array}{l} \quad \mathrm{data}[f_0 \mapsto \mathrm{id}, [\mathrm{name} = \mathrm{id}]] \\ \| \ \mathrm{customer}[f_1 \mapsto \mathrm{bookingref}_{\langle d,\mathrm{id}\rangle}, [\ldots]] \\ \| \ \mathrm{broker}[f_2 \mapsto \mathrm{bookingref}_{\langle d,\mathrm{id}\rangle}, [\ldots]] \\ \| \ \mathrm{hotel}[f_3 \mapsto \mathrm{bookingref}_{\langle d,\mathrm{id}\rangle}, [\ldots]] \end{array}\right) \setminus \mathrm{data.name} =$$

$$\begin{array}{l} \quad \mathrm{data}[f_0 \mapsto \mathrm{name}, [\mathrm{name} = \langle\rangle]] \\ \| \ \mathrm{customer}[f_1 \mapsto \mathrm{bookingref}_{\langle d,\mathrm{id}\rangle}, [\ldots]] \\ \| \ \mathrm{broker}[f_2 \mapsto \mathrm{bookingref}_{\langle d,\mathrm{id}\rangle}, [\ldots]] \\ \| \ \mathrm{hotel}[f_3 \mapsto \mathrm{bookingref}_{\langle d,\mathrm{id}\rangle}, [\ldots]] \end{array}$$

In this example program, hiding data.name is not preserved by the program evaluation. The program thus does not meet the prerequisites of Theorem 1. It is also intuitively clear, that this program is *not* secure as the broker has access to the private booking result of the customer.

Example with Currying

By contrast, in the program that uses the curried version, hiding is respected by the program evaluation and therefore secure according to our Theorem 1.

$$t \equiv \begin{array}{l} \quad \mathrm{data}[\varnothing, [\mathrm{name} = \mathrm{id}]] \\ \| \ \mathrm{customer}[f_0 \mapsto \mathrm{broker.book}_C(d).\mathrm{room'}(\mathrm{data.name}), t] \\ \| \ \mathrm{broker}[\varnothing, [\mathrm{book}_C = \varsigma(x,d)\mathrm{hotel.room}_C(d), \ldots]] \\ \| \ \mathrm{hotel}[\varnothing, [\mathrm{room} = \varsigma(x,(d,n))\mathrm{bookingref}, \\ \qquad\qquad \mathrm{room}_C = \varsigma(x,d)[\mathrm{room'} = \varsigma(x,n)x.\mathrm{room}(d,n)]]] \end{array}$$

This configuration reduces to the following, where H abbreviates the active object of hotel.

$$\begin{array}{l} \quad \mathrm{data}[f_1 \mapsto \mathrm{id}, [\mathrm{name} = \mathrm{id}]] \\ \| \ \mathrm{customer}[f_0 \mapsto f_3.\mathrm{room'}(\mathrm{id}), t] \\ \| \ \mathrm{broker}[f_2 \mapsto f_3, [\ldots]] \\ \| \ \mathrm{hotel}[f_3 \mapsto [\mathrm{room'} = \varsigma(x,n)\mathrm{H.room}(d,n)], [\ldots]]] \end{array}$$

Replying the semi-evaluated function via f_3 to customer and broker and reducing locally we get the personalized bookingref in customer's request list.

$$t' \equiv \begin{array}{l} \quad \mathrm{data}[f_1 \mapsto \mathrm{id}, [\mathrm{name} = \mathrm{id}]] \\ \| \ \mathrm{customer}[f_0 \mapsto \mathrm{bookingref}_{\langle d,\mathrm{id}\rangle}), t] \\ \| \ \mathrm{broker}[f_2 \mapsto [\mathrm{room'} = \varsigma(x,n)\mathrm{H.room}(d,n)], [\ldots]] \\ \| \ \mathrm{hotel}[f_3 \mapsto [\mathrm{room'} = \varsigma(x,n)\mathrm{H.room}(d,n)], [\ldots]]] \end{array}$$

The crucial test is now to apply hiding first to the initial configuration t'.

$$\left(\begin{array}{l} \quad \mathrm{data}[\varnothing, [\mathrm{name} = \mathrm{id}]] \\ \| \ \mathrm{customer}[f_0 \mapsto \mathrm{broker.book}_C(d).\mathrm{room'}(\mathrm{data.name}), t] \\ \| \ \mathrm{broker}[\varnothing, [\mathrm{book}_C = \varsigma(x,d)\mathrm{hotel.room}_C(d), \ldots]] \\ \| \ \mathrm{hotel}[\varnothing, [\mathrm{room} = \varsigma(x,(d,n))\mathrm{bookingref}, \\ \qquad\qquad \mathrm{room}_C = \varsigma(x,d)[\mathrm{room'} = \varsigma(x,n)x.\mathrm{room}(d,n)]]] \end{array}\right) \setminus \mathrm{data.name} =$$

$$\begin{array}{l} \quad \mathrm{data}[\varnothing, [\mathrm{name} = \langle\rangle]] \\ \| \ \mathrm{customer}[f_0 \mapsto \mathrm{broker.book}_C(d).\mathrm{room'}(\mathrm{data.name}), t] \\ \| \ \mathrm{broker}[\varnothing, [\mathrm{book}_C = \varsigma(x,d)\mathrm{hotel.room}_C(d), \ldots]] \\ \| \ \mathrm{hotel}[\varnothing, [\mathrm{room} = \varsigma(x,(d,n))\mathrm{bookingref}, \\ \qquad\qquad \mathrm{room}_C = \varsigma(x,d)[\mathrm{room'} = \varsigma(x,n)x.\mathrm{room}(d,n)]]] \end{array}$$

Reducing this, we reach the following configuration t^{Δ}.

$$t^{\Delta} \equiv \begin{array}{l} \quad \mathrm{data}[f_1 \mapsto \langle\rangle, [\mathrm{name} = \langle\rangle]] \\ \| \ \mathrm{customer}[f_0 \mapsto \mathrm{bookingref}_{\langle d,\langle\rangle\rangle}), t] \\ \| \ \mathrm{broker}[f_2 \mapsto [\mathrm{room'} = \varsigma(x,n)\mathrm{H.room}(d,n)], [\ldots]] \\ \| \ \mathrm{hotel}[f_3 \mapsto [\mathrm{room'} = \varsigma(x,n)\mathrm{H.room}(d,n)], [\ldots]]] \end{array}$$

Now, if we consider hiding Δ in the reduction of the previous configuration, i.e. $t' \setminus \Delta$ we get this configuration.

$$t' \setminus \mathrm{data.name} \equiv \begin{array}{l} \quad \mathrm{data}[f_1 \mapsto \langle\rangle, [\mathrm{name} = \langle\rangle]] \\ \| \ \mathrm{customer}[f_0 \mapsto \mathrm{bookingref}_{\langle d,\mathrm{id}\rangle}), t] \\ \| \ \mathrm{broker}[f_2 \mapsto [\mathrm{room'} = \varsigma(x,n)\mathrm{H.room}(d,n)], [\ldots]] \\ \| \ \mathrm{hotel}[f_3 \mapsto [\mathrm{room'} = \varsigma(x,n)\mathrm{H.room}(d,n)], [\ldots]]] \end{array}$$

At first sight, this seems odd because $t^{\Delta} \neq t \setminus \Delta$ (they differ in f_0) but we have $t^{\Delta} =_{\nabla} t \setminus \Delta$. This suffices for the conclusion of Theorem 1. We have thus shown that this program is secure for Δ.

5 Conclusions

5.1 Related Work

Myers and Liskow augmented the DLM model with the idea of information flow control as described in the papers [ML00]. Further works by Myers have been mostly practically oriented, foundations only considered later [ZM07]. Initially, he implemented a Java tool package called JFlow, nowadays JIF, that implements his Decentralzied Label Model (DLM) based on information flow control [Mye99]. In more recent work, still along the same lines, Zheng and Myers [ZM07]

have gone even further in exploring the possibilities to dynamically assign security labels while still trying to arrive at static information flow control. The main criticism to the DLM is that it *assumes that all principals respect the DLM*. We also consider this as a weakness in particular in distributed applications where assumptions about remote parties seems inappropriate. To illustrate this difference: in our example above the DLM would have assumed that the customer's call of book to the broker would also be high and thus be treated confidentially. Contrarily to this strong assumption of the DLM, we do *not make any assumptions about the low site*. In particular the customer can see everything in his request queue, be it marked high or low.

One very popular strand of research towards verification of security has been static analysis of noninterference using type checking. Briefly, this means that type systems are constructed that encode a noninterference property. When a program passes the type check for that system then we know that it has a certain security type. The security type can be for example an assignment of program data to security classes. The type check then guarantees that the information flows are only those allowed by the assignment of the data to the security classes. A good survey giving an introduction to the matter and comparing various activities is given by Sabelfeld and Myers [SM03]. A constructive way to support noninterference analysis is by providing a set of basic properties that can be combined to build various forms of noninterference. H. Mantel's work since his PhD has mainly focused on providing such a logical tool box [Man00, Man02].

Focardi and Gorrieri's work has strongly influenced our current approach. We take their method of algebraic characterization of information flow security further in applying them to the distributed object calculus ASP_{fun} thereby addressing real language issues.

5.2 Discussion and Outlook

We aim at using functional active objects as a language calculus and build a logical tool set for compositional properties. Since already on the simpler trace models the definition and theory development for a MAKS is an intrinsically complex task, we additionally employ Isabelle/HOL as a verification environment to support us. The derived LB-MAKS security properties shall be transformed into security type systems – in the sense as described in the previous section – to derive practically useful static analysis tools from our mechanized LB-MAKS. The presented currying concept is a first important step towards such an enforcement library. A further important stepping stone is the definition of hiding for ASP_{fun}. It helps bridging the gap between abstract trace based semantics and more detailed language based models as illustrated in this paper.

To our knowledge no one has considered the use of futures in combination with confinement given by objects as a means to characterize information flow. The major advantage of our approach is that we are less abstract than event systems while being abstract enough to consider realistic distributed applications. Currying for flexible parameterization can be applied to other languages, in particular to functional ones, to support privacy in distributed applications.

References

[AC96] Abadi, M., Cardelli, L.: A Theory of Objects. Springer, New York (1996)

[BNRNP08] Bauer, J., Nielson, F., Riis-Nielson, H., Pilegaard, H.: Relational analysis of correlation. In: Alpuente, M., Vidal, G. (eds.) SAS 2008. LNCS, vol. 5079, pp. 32–46. Springer, Heidelberg (2008)

[CH05] Caromel, D., Henrio, L.: A Theory of Distributed Objects. Springer, New York (2005)

[FG95] Focardi, R., Gorrieri, R.: A classification of security properties for process algebras. Journal of Computer Security 3(1), 5–33 (1995)

[FK10] Fleck, A., Kammüller, F.: Implementing privacy with erlang active objects. In: 5th International Conference on Internet Monitoring and Protection, ICIMP 2010. IEEE, Los Alamitos (2010)

[HK09] Henrio, L., Kammüller, F.: Functional active objects: Typing and formalisation. In: 8th International Workshop on the Foundations of Coordination Languages and Software Architectures, FOCLASA 2009. ENTCS. Elsevier, Amsterdam (2009); Also invited for journal publication in Science of Computer Programming, Elsevier

[Kam10] Kammüller, F.: Using functional active objects to enforce privacy. In: 5th Conf. on Network Architectures and Information Systems Security, SAR-SSI 2010 (2010)

[Man00] Mantel, H.: Possibilistic definitions of security – an assembly kit. In: Computer Security Foundations Workshop, pp. 185–199. IEEE, Los Alamitos (2000)

[Man02] Mantel, H.: On the composition of secure systems. In: Symposium on Security and Privacy (2002)

[ML00] Myers, A.C., Liskov, B.: Protecting privacy using the decentralized label model. ACM Transactions on Software Engineering and Methodology 9, 410–442 (2000)

[Mye99] Myers, A.C.: Jflow: Practical mostly-static information flow control. In: 26th ACM Symposium on Principles of Programming Languages, POPL 1999 (1999)

[Pro08] ProActive API and environment (2008), `http://www.inria.fr/oasis/proactive` (under LGPL)

[SM03] Sabelfeld, A., Myers, A.C.: Language-based information-flow security. Selected Areas in Communications 21, 5–19 (2003)

[ZM07] Zheng, L., Myers, A.C.: Dynamic security labels and static information flow control. International Journal of Information Security 6(2-3) (2007)

L–PEP: A Logic to Reason about Privacy–Enhancing Cryptography Protocols

Almudena Alcaide[1], Ali E. Abdallah[2],
Ana I. González–Tablas[1], and José M. de Fuentes[1]

[1] Department of Computer Science,
University Carlos II of Madrid, Spain
{aalcaide,aigonzal,jfuentes}@inf.uc3m.es
[2] Institute for Computing Research,
London South Bank University, UK
a.abdallah@lsbu.ac.uk

Abstract. In recent years, many cryptography protocols have been designed for many different scenarios, with the purpose of preserving security of communications as well as privacy and anonymity of participant entities. In general, every proposed solution has possed a real challenge to the existing formal methods of protocol analysis and verification. The main goal of this work is the proposal of a logic to reason about privacy-enhancing monotonic and non–monotonic cryptography protocols. The new logic will be called L-PEP and it extends the existing Rubin's logic of beliefs.

1 Introduction

This work has been mainly motivated by the following two factors. On the one hand, organizations are increasingly introducing privacy preserving mechanisms to comply with laws and regulations, and to satisfy users' and employees' privacy requirements. As a result, newly designed system architectures and security protocols require of additional privacy and security features such as anonymity and pseudonymity, unlinkability and unobservability of transactions, selective disclosure of information, privacy policies embedded in data, etc. In particular, the set of building block primitives used in privacy–enhancing cryptography protocols include anonymous credentials, blind signatures, dual signatures, recursive hashing, zero–knowledge proofs, identity–based encryption, anonymous biometrics, etc. As it happens, in every occasion the proposed cryptographic solution has possed a real challenge to the existing formal methods of protocol analysis and verification as none of the existing methods include the necessary inference rules to formally reason about those operations [1]. To this regard, the formal modeling of protocol privacy requirements represents a significant step forward, as it will allow us to validate protocol privacy properties, in the same way as it is done with security properties such us confidentiality, integrity or accessibility. So far, not very much work has been done in this direction.

J. Garcia-Alfaro et al. (Eds.): DPM 2010 and SETOP 2010, LNCS 6514, pp. 108–122, 2011.

On the other hand, privacy–enhancing technologies have placed the emphasis on determining the so called *Private Data Life Cycle*. Private data life cycle starts when data is created and finishes when the data is destroyed. The aim of those new technologies and the security protocols deployed to such effect is to control the whole cycle, including implicit and explicit disclosures of information, deletion of data, custody constraints, etc. For example, a common privacy constraint describes users' requirements to determine how long for entities are entitled to use or store someone else's private information, which might have been received in one of the steps of a cryptographic protocol. That is, via privacy policies and agreements, users could force other entities to delete and destroy users' own personal data after a fixed amount of time or, if data are not further required for any of the purposes which they were stored for [2]. Formally, a *non–monotonic* security protocol refers to the non–incremental acquisition of knowledge by participants along the execution of a security protocol. In fact, participant entities might have to *forget* information or data which they once knew or held, during a particular protocol instance. Formal representation and verification of non–monotonic schemes is pending a global solution.

1.1 Validation Logics

Since the design of BAN logic by Burrows, Abadi and Needham [3], many derivatives of BAN, as well as new logics, have been defined to formally and monotonically reason about cryptographic protocols. They are also known as *Logics of Beliefs*, as protocol goal verification is based on the set of beliefs and possessions that entities hold at the end of a protocol execution (for example, entity A *believes* K_{AB} is a secret key to communicate with entity B). In this context, monotonicity refers to the way in which the knowledge and the beliefs of a particular entity increase as the protocol execution progresses. In a monotonic protocol, once something is known, it is known forever.

By contrast, a non–monotonic logic would allow us to reason about non–monotonic protocols, in which entities might *stop believing* in a piece of information, or *forget* knowledge during the execution of a protocol. This is, in fact, the type of reasoning it is needed for the formal verification of many of today's privacy–enhancing cryptography protocols which rely on the deletion of information. Only a few attempts have been made to elaborate on this type of reasoning [4–6]. In [4] authors define a *not* operator to construct negation. In [5, 6] authors include actions to directly delete beliefs from entities' belief sets. We have chosen Rubin's logic [6] to be the focus of our study. The main reasons being that Rubin's logic is more tractable than Moser's [5] and furthermore, it allows us to differentiate between non–monotonicity of knowledge and non–monotonicity of belief. In addition, Rubin's logic does not require protocol *idealization* and the notation and the reasoning are similar to the original BAN logic [3].

1.2 Overview of Our Work

The main goal of this work is the proposal of a logic to reason about privacy-enhancing cryptographic protocols. The new logic will be called L-PEP and it extends Rubin's logic of beliefs [6]. Rubin's logic is extended by adding new inference rules and actions and by enhancing some of the original actions and rules. L-PEP, as presented in this paper, will allow us to formally reason about cryptographic hashing, recursive cryptographic hashing, concatenation of hashes and dual signatures, all these being primitives of existing privacy–enhancing cryptography protocols. In this paper, and to illustrate how the formalism works, we formally analyze two different protocols. Firstly, we apply the new L–PEP Logic to the analysis of the dual signature mechanism defined in SET (Secure Electronic Transaction protocol) [7]. Secondly, we will formally reason about a novel *non–monotonic* protocol applied to the electronic speed ticketing in vehicular ad–hoc networks (VANETs). We will give a brief description of this system and analyze vehicular privacy related goals.

Previous related work can be found in [8–10]. In [8] and [9] authors use Rubin's Logic to formally analyze an e–commerce protocol and two authentication protocols respectively. However, the analysis is monotonic (i.e. only incremental knowledge is considered) and no signatures applied to hashed values (signatures with appendix) are formally considered during the protocols' formal analysis. Finally, in [10], a non–monotonic authenticated routing protocol is analyzed in which entities must *forget* other participant's public key certificates once these are revoked by the corresponding authorities. In that work, neither privacy related goals nor signatures applied to hashed values are formally analyzed.

The rest of the paper is organized as follows. In Section 2 we describe Rubin's logic main components including the additional features and enhancements proposed in this work. In Section 3 we illustrate the new formalism giving formal proof of privacy related goals regarding the Dual signature mechanism in the SET protocol. In Section 4 we illustrate the new formalism giving formal verification of a vehicular speed ticketing electronic protocol. Finally, Section 5 is devoted to establish final considerations and future work.

2 Rubin's Logic and Enhancements

In this section we present an extension as well as few enhancements to the non–monotonic logic introduced by Rubin in [6]. Some of the concepts and notation have been synthesized, please refer to [3] and [6] for a detailed description.

2.1 Non–monotonicity of Knowledge vs. Non–monotonicity of Belief

Rubin introduced a logic in which it is easy to difference between non–monotonicity of knowledge and non–monotonicity of belief in the following way: while entities *know* things such as keys, nonces, or data, entities *believe* in metadata, such as freshness of data, or in other entities' possessions. When a principal

stops believing in something it does not imply it also stops possessing such an item or knowing the data. By contrast, if an entity stops possessing or knowing an item, it does automatically stop believing in it as it does not longer hold it. This fact is very relevant in today's schemes. Principals are forced, via privacy agreements, to *forget* (stop storing and destroy) certain information and some protocols base their correctness on that premise. Furthermore, *forgetting* information implies readjustments in other principals' sets of beliefs. For example, a merchant *forgets* the user's card details after a transaction is processed, and the user *stops believing* that the merchant possesses such information.

Special notation

$\sharp X$	Freshness of item X.
X *contains* x	Denotes x is part of item X.
$:=$	Denotes assignment.
$k_{P_i}^{-1}$	Denotes entity P_i private key.
k_{P_i}	Denotes entity P_i public key.
$\{X\}_k$	Denotes token X encrypted with key K.

2.2 Global Definitions

The specification of a protocol is the starting point of the analysis.

- **Set of participants:** W denotes the whole *World* of entities. $P = \{P_1, \ldots, P_n\} \subseteq W$ denotes the set of entities participating in the protocol.
- **Set of inference rules:** R denotes the set of inference rules. Each entity applies these rules to its sets of beliefs and knowledge.
- **Set of secrets:** S denotes the set of secrets. Data which is meant to be kept secret from the *World*. For each secret $s \in S$ the set ***Observers(s)*** denotes the set of entities which know secret s. Every communication is assumed to be listened by the whole *World* so after each communication, a function ***Update*** will recursively update the appropriate *Observers* sets. The Update functions also operates over the set of possessions of each entity.
- **TRUST Matrix :** A matrix TRUST is used to represent the trust relationship between every pair of principals. An entity P_i is said to trust entity P_j if P_i behaves as though P_j will follow the protocol.
- **Free variables:** FV denotes the set *free* variables to be instantiated along the formal proof.
- **Bound variables:** BV denotes the set *bound* variables which must hold the same value throughout the whole formal proof.

2.3 Definitions Local to Each Protocol Participant

The following local sets are private to each protocol participant.

- **Set of Possessions:** $POSS(P_i)$ denotes the possessions of participant P_i. For example: $POSS(P_i) = \{X, k_{P_i}^{-1}\}$. This is, P_i possesses (*knows*) X and private key $k_{P_i}^{-1}$.

- **Set of Beliefs:** $BEL(P_i)$ denotes the beliefs of participant P_i. For example: $BEL(P_i) = \{\sharp k, \{k_{P_j}^{-1}, k\} \in POSS(P_j)\}$. This is, P_i believes that k is a *fresh* key and that P_j is in possession of private key $k_{P_j}^{-1}$ and secret key k.
- **Behavior List:** $BL(P_i)$ denotes the behavior list of principal P_i. This list dictates to entity P_i what actions to take along the protocol execution and in which order. For example: $BL(P_i) = \langle bvr_1, brv_2, \ldots, bvr_n \rangle$. Where each bvr_i is of the form: $bvr_i = \{msg.operation; AL\}$. Where:
  - $msg.operation$ denotes one of the following operations: $send(P_j, m)$ or $receive(P_j, m)$, to denote sending a message m to P_j or receiving a message m from P_j, respectively. After a $receive(P_j, m)$ message operation is performed the corresponding $POSS$ sets are modified accordingly. After a *send* message operation is performed the ***Update*** function is recursively called to modify the appropriate *Observers* sets.
  - AL denotes an ordered list of zero or more *actions* that participants have to perform along the protocol execution. These actions are described in detail in section 2.4.

2.4 List of Actions

Describe how entities perform operations over data. All variables involved in these formula are considered to be bound variables. Only those actions related to our scope are listed and only the new ones are described in detail. Please refer to [6] for a complete list.

A1. **Apply**(f, X, n)
Condition: $\{f, X\} \subset POSS(P)$
Result: $POSS(P_i) := POSS(P_i), \cup\{f^l(X)\ \forall l \leq n\}$
Description: This new action is used when P_i applies the function f to X in a recursive way. Function f could be a hash function, a polynomial, a biometric template, a boolean function, XOR, etc.

A2. **Concat**(x_1, x_2)
Condition: $\{x_1, x_2\} \subset POSS(P_i)$
Result: $POSS(P_i) := POSS(P_i) \cup \{(x_1 \| x_2)\}$

A3. **Decrypt**$(\{X\}_k, k)$
Condition: $\{\{X\}_k, k\} \subset POSS(P_i), P_i \in Observers(k)$
Result: $POSS(P_i) := POSS(P_i) \cup \{X\}$

A4. **Decrypt–asymm**$(\{X\}_{k_{P_j}}; k_{P_j}^{-1})$
Condition: $\{\{X\}_{k_{P_j}}; k_{P_j}^{-1}\} \subset POSS(P_j),\ P_j \in Observers(k_{P_j}^{-1})$
Result: $POSS(P_j) := POSS(P_j) \cup \{X\}$
Description: This new action is used when a principal decrypts an encrypted item with the corresponding private key.

A5. **Decrypt–signature**$(\{X\}_{k_{P_j}^{-1}}; k_{P_j})$
Condition: $\{\{X\}_{k_{P_j}^{-1}}, k_{P_j}\} \subset POSS(P_i),\ P_i \in Observers(k_{P_j})$
Result: $POSS(P_i) := POSS(P_i) \cup \{X\},\ BEL(P_i) := BEL(P_i) \cup \{X \in POSS(P_j)\}$

Description: This new action is used when a principal decrypts an encrypted item with the corresponding public key. Note that this rule refers to *message recovery* signature schemes.

A6. **Encrypt**(X; k)
Condition: $\{X, k\} \subset POSS(P_i), P_i \in Observers(k)$
Result: $POSS(P_i) := POSS(P_i) \cup \{X\}_k$

A7. **Forget**(X)
Condition: $X \in POSS(P_i)$
Result: $POSS(P_i) := POSS(P_i) - \{X\}$, $BEL(P_i) := BEL(P_i) - \{\sharp X\}$, $\forall j$ *such that* $TRUST[j,i] = 1$ then $BEL(P_j) := BEL(P_j) - \{X \in POSS(P_i)\}$

A8. **Forget–secret**(s)
Condition: $P_i \in Observers(s)$, $s \in POSS(P_i)$
Result: $POSS(P_i) := POSS(P_i) - \{s\}$, $BEL(P_i) := BEL(P_i) - \{\sharp s\}$, $Observers(s) := Observers(s) - \{P_i\}$, $\forall j$ *such that* $TRUST[j,i] = 1$ then $BEL(P_j) := BEL(P_j) - \{s \in POSS(P_i)\}$

A9. **Generate–secret(s)**
Result: $S := S \cup \{s\}$, $Observers(s) := \{P_i\}$, $POSS(P_i) := POSS(P_i) \cup \{s\}$, $BEL(Pi) := BEL(Pi) \cup \{\#(s)\}$

A10. **Split**($\{(x_1 \| x_2)\}$)
Condition: $\{(x_1 \| x_2)\} \in POSS(P_i)$
Result: $POSS(P_i) := POSS(P_i) \cup \{x_1, x_2\}$

2.5 Set of Inference Rules

After each action, each participant updates its BEL set using all possible inference rules. Note that, rules are used to propagate belief during the protocol run whereas actions are used to propagate knowledge. We will now describe the most relevant rules for the scope of our work. Only the new ones are fully described.

R1. **Function Rule**
Condition: $\{\{f, X\} \subset POSS(P_i)\} \in BEL(P_j)$
Result: $\{Apply(f, X, n) \in POSS(P_i)\} \in BEL(P_j)$
Description: This new rule states that if P_j believes that P_i possesses f and X then, P_j believes that P_i possesses $f^i(X)$ $\forall 1 \leq i \leq n$

R2. **Message Meaning Rule**
Condition: $\{X\}_k \in POSS(P_i)$, $\{P_i, P_j\} \subseteq Observers(k)$
Result: $BEL(P_i) := BEL(P_i) \cup \{X \in POSS(P_j)\}$

R3. **Signature Verification Rule**
Condition: $\{\{h(X)\}_{k_{P_j}^{-1}}, X\} \in POSS(P_i)$, $\{h(X) \in POSS(P_j)\} \in BEL(P_i)$
Result: $BEL(P_i) := BEL(P_i) \cup \{X \in POSS(P_j)\}$
Description: Although asymmetric encryption has been considered in formal logic reasoning since the original BAN logic, signature verification has not always been properly addressed. It was only very recently (2009) that it was noted that the *hashing inference rule* originally defined in

BAN was faulted [11]. The rule had been used to proof correctness of many schemes, for example, the SET protocol [12]. The same authors propose some new hash inference rules to add to BAN logic, however, these rules enforce sender authentication for every hash value received. By contrast, we propose a new signature verification rule such that we are able to link a signed hash value to its original value by authenticating the signer entity of the hash. Indeed, the expression $\{h(X) \in POSS(P_j)\} \in BEL(P_i)$ is true if and only if the action Decrypt–signature$(\{h(X)\}_{k_{P_j}^{-1}})$ has been applied successfully.

R4. **Split Rule**

Condition: $\{f, f(x_1) \parallel f(x_2)\} \in POSS(P_i)$

Result: $\{f(x_1), f(x_2)\} \in POSS(P_i)$

Description: This rule states that if P_i knows function f, then it can split concatenated items. This is common as knowing a function will give us information about the length of its output.

2.6 Formal Verification Process

The verification process is a recursive, automated process in which inference rules and the *Update* function are recursively applied until none of the rules can proceed any further and the *Observer* sets cannot be modified. Note that the processes described below only highlight the most relevant steps of the verification process.

3 SET's Dual Signature

To illustrate how the formalism described in previous sections works, and how it can be used to formally reason about privacy properties, we introduce the dual signature mechanism used in the SET protocol [7] for e-commerce transactions. The goal of the dual signature mechanism is to implement a *minimum information disclosure* procedure. As previously mentioned, although various attempts have been made to formally reason about SET [12], the hash inference rule applied was not correct [11].

Notation:

$\{U, M, B\}$	Denotes the set of participant entities: User, Merchant and Bank.
K_U^{-1}	Denotes user U private key.
K_{UM}	Denotes a secret key between the user U and the merchant M.
K_{UB}	Denotes a secret key between the user U and the bank B.
h	Cryptographic hash function.

– **Step 1:** User U wants to proceed with an electronic purchase. User U constructs an *Order Information* token denoted as OI describing the concept of the purchase, quantity, price, etc. Additionally, U also generates a *Payment*

Information token denoted by PI, including the card details and the amount to be paid. Item OI is destined to the Merchant M and item PI is destined to the Bank B. Both items must be linked as the user must be able to prove that the payment is intended for the corresponding order, however, OI must be kept secret from B as well as PI must be kept secret from the merchant M.

- **Step 2:** User U sends Merchant M the following message:

$$m_1 = \{OI, h(PI), \{h(h(OI)\|h(PI))\}_{K_U^{-1}}\}_{K_{UM}}$$

In receiving this item, entity M can verify the user's signature over item OI and can also establish the link between the OI and PI without getting to know the content of PI.

- **Step 3:** User U sends the Bank B the following message:

$$m_2 = \{h(OI), PI, \{h(h(OI)\|h(PI))\}_{K_U^{-1}}\}_{K_{UB}}$$

In receiving this item, entity B can verify the user's signature over item PI and can also establish the link between the OI and PI without getting to know the content of OI.

3.1 SET's Dual Signature Formal Verification

As the formal reasoning of steps 2 and 3 are equivalent, we will only proceed with the formal verification of steps 1 and 2.

Global and local definitions:

- W, the world.
- $P = \{U, B, M\}$, is the set of participant entities.
- $S = \{K_{UM}, K_{UB}, K_U, K_U^{-1}\}$, is the set of secrets. Where $Observers(K_{UM}) = \{U, M\}$; $Observers(K_{UB}) = \{U, B\}$; $Observers(K_U) = \{W\}$; $Observers(K_U^{-1}) = \{U\}$;
- $R = \{R1, R2, R3, R4\}$ is the set of inference rules.
- $\forall i, j \in \{1, 2, 3\}, TRUST_{ij} = 0$
- $FV = \{x_1, x_2, x_3\}$
- $BV = \{h, OI, PI\}$. Where h represents a cryptographic hash function, OI the order information and PI the payment information.

Principal U's local sets:

$POSS(U) = \{K_{UM}, K_{UB}, K_U^{-1}, K_U, h\}$
$BEL(U) = \{\sharp K_{UM}, \sharp K_{UB}\}$
$BL(U) = \langle bvr_1^U, bvr_2^U \rangle$
Where:
$bvr_1^U = \{\ -\ ; generate\_secret(OI), generate\_secret(PI)\}$
$bvr_2^U = \{Send(M, m_1);\ -\ \}$

Principal M's local sets:

$POSS(M) = \{K_U, K_{UM}, h\}$
$BEL(M) = \{\sharp K_{UM}, \sharp K_U, \{K_U^{-1}, h\} \in POSS(U)\}$
$BL(M) = \langle bvr_1^M \rangle$
Where:
$bvr_1^M =$ $\{Receive(U, m);$
$Decrypt(m, K_{UM})$ into x_1, x_2, x_3
$Apply(h, x_1, 1),$
$Concat(h(x_1), x_2),$
$Decrypt - signature(x_3, K_U)\}$

Note that entity M would believe that $\{K_U^{-1} \in POSS(U)\}$ and $\sharp K_U$ once the corresponding public key certificate has been properly verified.

Expected security and privacy goals:

G1: $\{OI\} \in POSS(M)$. M possesses item OI
G2: Observers(OI)={U,M} . Secret OI must only be known by U and M
G3: $\{OI \in POSS(U)\} \in BEL(M)$. M must believe that U possesses OI
G4: $\{\{h(OI) \| h(PI)\} \in POSS(U)\} \in BEL(M)$. M must believe that U possesses $\{h(OI) \| h(PI)\}$
G5: Observers(PI)={U,B} . Secret PI must only be known by U and B

Formal proof: Only those steps in relation to the satisfaction of goals are shown.

Step 1:
$bvr_1^U = \{ \ - \ ; generate\_secret(OI), generate\_secret(PI)\}$
$S := S \cup \{OI, PI\}$
$Observers(OI) = \{U\}$
$Observers(PI) = \{U\}$
$bvr_2^U = \{Send(M, m_1); \ - \ \}$
Step 2:
$bvr_1^M =$
$\{$**Receive**$(U, m_1); \ - \ \}$
$POSS(M) := POSS(M) \cup \{m_1\}$
Decrypt$(\{OI, h(PI), \{h(h(OI) \| h(PI))\}_{K_U^{-1}}\}\}_{K_{UM}}, K_{UM}\})$
$POSS(M) := POSS(M) \cup \{x_1 = OI, x_2 = h(PI),$
$x_3 = \{h(h(OI) \| h(PI))\}_{K_U^{-1}}\}\}$ **(G1)**
Message Meaning Rule R2:
$BEL(M) := BEL(M) \cup \{x_1 = OI, x_2 = h(PI),$
$x_3 = \{h(h(OI) \| h(PI))\}_{K_U^{-1}}\} \in POSS(U)\}$
(G3)
Function Rule R1 for hash h and *concat* functions:
$BEL(M) := BEL(M) \cup \{h(h(OI) \| h(PI)) \in POSS(U)\}$
Apply$(h, x_1, 1)$
$POSS(M) := POSS(M) \cup \{h(OI)\}$

Concat$(h(x_1), x_2)$
$POSS(M) := POSS(M) \cup \{h(OI)\|h(PI)\}$
Decrypt-signature$(x_3 = \{h(h(OI)\|h(PI))\}_{K_U^{-1}}\}, K_U)$
$POSS(M) := POSS(M) \cup \{h(h(OI)\|h(PI))\}$,
$BEL(M) := BEL(M) \cup \{\{h(h(OI)\|h(PI))\} \in POSS(U)\}$
Signature Verification rule R3:
$BEL(M) := BEL(M) \cup \{\{h(OI)\|h(PI)\} \in POSS(U)\}$**(G4)**

The ***Update*** function, recursively applied until the end of the verification process, renders the following results over the secrets OI and PI:
$Observers(OI) = \{U, M\}$ **(G2)**
$Observers(PI) = \{U, B\}$ **(G5)**

□

4 Private Speed Ticketing Electronic System

Vehicular ad–hoc networks (VANETs) allow nodes, including vehicles and road side units, to communicate with each other. These networks are used with the purpose of optimizing traffic (reducing congestion and accidents), obtaining real–time road information (the vehicles can serve as information collectors) and giving authorities the possibility to supervise vehicles electronically (speed control, vehicular permits, etc). In VANETs, vehicles are equipped with tamper proof devices known as On–Board Units (OBUs) used to communicate with other vehicles and with other Road–Side Units (RSUs) part of the road infrastructure. In VANETs, vehicles and drivers ought to maintain their identities private, at the same time as they must keep accountable for their actions and, the information processed in the VANET must be totally reliable. In this section we sketch a speed ticketing electronic system which allows the transportation authorities to enforce speed limits over a stretch of road, at the same time as the identity of the non–offenders vehicles is kept private[1].

The system consists of :

- A Ticketing Authority (TA) responsible for managing unique vehicle identifiers, as for example the Electronic Chassis Number (ECN).
- On–Board Units (OBUs). An OBU is a tamper proof device included in each vehicle on the road. The ECN of a vehicle is inserted in the vehicle's OBU such that when a vehicle receives a $\{request\_id\}$ token, enquiring about its identity, the OBU responds with the ECN encrypted.
- Road–Side Units (RSUs) which periodically broadcast messages over *Dedicated Short Range Channels* requesting the identity of the passing by vehicles. The RSUs compose *sp_evidence* tokens from the responses received from the passing vehicles by attaching the time to the vehicle identification token. This new token called *sp_evidence* is sent to a Ticketing Server (TS).

[1] The system here described is being enhanced and developed further. For the subject of the scope of this article we only detail the most basic version.

Table 1. Speed Ticketing Electronic Protocol monitoring a stretch of road between road–side units RSU_1 and RSU_2

m_1. $RSU_1 \rightarrow World$:	$\{request\_id\}$
m_2. $OBU \rightarrow RSU_1$:	$\{ECN\}_{K_{TS}}$
m_3. $RSU_1 \rightarrow TS$:	$\{sp\_evidence_1\}_{K^{-1}_{RSU_1}}$
m_4. $RSU_2 \rightarrow World$:	$\{request\_id\}$
m_5. $OBU \rightarrow RSU_2$:	$\{ECN\}_{K_{TS}}$
m_6. $RSU_2 \rightarrow TS$:	$\{sp\_evidence_2\}_{K^{-1}_{RSU_2}}$
m_7. $TS \rightarrow TA$:	$\{sp\_ticket\}$
	Where:
	$sp\_evidence_1 = \{\{ECN\}_{K_{TS}} \parallel t_1\}$
	$sp\_evidence_2 = \{\{ECN\}_{K_{TS}} \parallel t_2\}$
	$sp\_ticket = \{TS \parallel ECN \parallel t_1 \parallel t_2\}_{K_{TA}}$

– The Ticketing Server (TS) receives *sp_evidence* tokens from two different RSUs and issue *sp_ticket* tickets for those vehicles which took less than the permitted time to cover a distance between the locations of the two RSUs. The TS is also responsible for informing the Ticketing Authority (TA) about which vehicles have over passed the speed limit over the stretch of road being monitored.

(Table 1 details the messages involved in the protocol.)

Although the messages exchanged between principals are detailed in Table 1, we will use the extension of Rubin's logic to formally represent each participant entity behavior list and the privacy related goals to be satisfied at the end of the message exchange.

4.1 Speed Ticketing Formal Verification

– **New action** Generate–ticket(ECN, t_1, t_2, t_l)
  Condition: $\{\{ECN\}_K, t_1, t_2, t_l\} \subset POSS(TS)$ such that: $(t_2 - t_1) < t_l$
  Result: $POSS(TS) := POSS(TS) \cup \{TS \parallel ECN \parallel t_1 \parallel t_2\}_{K_{TA}}$
  Description: This action is used to issue a *sp_ticket* when the difference between t_2 and t_1 is below a certain legal limit.
– **Global definitions:**
  - W, the world.
  - $P = \{TA, TS, RSU_1, RSU_2, OBU\}$, is the set of participant entities.
  - S=$\{K_{TA}, K_{TS}, K_{RSU_1}, K_{RSU_2}, ECN, K^{-1}_{TA}, K^{-1}_{TS}, K^{-1}_{RSU_1}, K^{-1}_{RSU_2}\}$, is the set of secrets. Where $Observers(K_{TA})$=$\{W\}$; $Observers(K_{TS})$=$\{W\}$; $Observers(K_{RSU_1}) = \{W\}$; $Observers(K_{RSU_2}) = \{W\}$; $Observers(ECN) = \{OBU\}$; $Observers(K^{-1}_{TA}) = \{TA\}$; $Observers(K^{-1}_{TS}) = \{TS\}$; $Observers(K^{-1}_{RSU_1}) = \{RSU_1\}$; $Observers(K^{-1}_{RSU_2}) = \{RSU_2\}$;
  - $R = \{R1, \ldots, R4\}$ is the set of inference rules.

- $\forall i, j \in \{1,2,3,4,5\}, TRUST_{ij} = 1$. Note that, unlike the previous example, all entities trust each other in following the protocol instructions. This is a reasonable assumption as all principals belong for the same authority, no tampering is possible on the OBU and the OBU trusts the authority responsible for the ticketing system.
- $BV = \{t_1, t_2, t_{limit}, sp\_evidence_1, sp\_evidence_2, sp\_ticket\}$. Where t_{limit} indicates the minimum legal time permitted in driving between units RSU_1 and RSU_2.

– **Principal** RSU_1**:**
$POSS(RSU_1) = \{K_{RSU_1}, K^{-1}_{RSU_1}\}$
$BEL(RSU_1) = \{\}$
$BL(RSU_1) = \langle bvr_1^{RSU_1}, bvr_2^{RSU_1}, bvr_3^{RSU_1}\rangle$
Where:
$bvr_1^{RSU_1} = \{Send(W, m_1);\ -\ \}$
$bvr_2^{RSU_1} = \{Receive(OBU, m_2);$
$Encrypt(Concat(m_2, t_1), K^{-1}_{RSU_1})\}$
$bvr_3^{RSU_1} = \{Send(TS, m_3);\ -\ \}$

– **Principal** RSU_2**:**
$POSS(RSU_2) = \{K_{RSU_2}, K^{-1}_{RSU_2}\}$
$BEL(RSU_2) = \{\}$
$BL(RSU_2) = \langle bvr_1^{RSU_2}, bvr_2^{RSU_2}, bvr_3^{RSU_2}\rangle$
Where:
$bvr_1^{RSU_2} = \{Send(W, m_4);\ -\ \}$
$bvr_2^{RSU_2} = \{Receive(OBU, m_5);$
$Encrypt(Concat(m_5, t_2), K^{-1}_{RSU_2})\}$
$bvr_3^{RSU_2} = \{Send(TS, m_6);\ -\ \}$

– **Principal** OBU**:**
$POSS(OBU) = \{ECN, K_{TS}\}$
$BEL(OBU) = \{\sharp K_{TS}, K^{-1}_{TS} \in POSS(TS)\}$. Note that these believes indicate that the vehicle's OBU has got a valid public key certificate from TS.
$BL(OBU) = \langle bvr_1^{OBU}, bvr_2^{OBU}, bvr_3^{OBU}, bvr_4^{OBU}\rangle$
Where:
$bvr_1^{OBU} = \{Receive(RSU_1, m_1); Encrypt(ECN, K_{TS})\}$
$bvr_2^{OBU} = \{Send(RSU_1, m_2);\ -\ \}$
$bvr_3^{OBU} = \{Receive(RSU_2, m_4); Encrypt(ECN, K_{TS})\}$
$bvr_4^{OBU} = \{Send(RSU_2, m_5);\ -\ \}$

– **Principal** TS**:**
$POSS(TS) = \{K_{TS}, K^{-1}_{TS}, K_{RSU_1}, K_{RSU_2}, t_{limit}\}$
$BEL(TS) = \{\sharp K_{RSU_1}, \sharp K_{RSU_2}, K^{-1}_{RSU_1} \in POSS(RSU_1),$
$K^{-1}_{RSU_2} \in POSS(RSU_2)\}.$
$BL(TS) = \langle bvr_1^{TS}, bvr_2^{TS}, bvr_3^{TS}\rangle$

Where:

$bvr_1^{TS} = \{Receive(RSU_1, m_3); \quad - \ \}$

$bvr_2^{TS} = \{Receive(RSU_2, m_6);$
$Decrypt - signature(m_3, K_{RSU_1}),$
$\textbf{\textit{Forget}}(\{sp\_evidence_1\}_{K^{-1}_{RSU_1}}),$
$Decrypt - signature(m_6, K_{RSU_2}),$
$\textbf{\textit{Forget}}(\{sp\_evidence_2\}_{K^{-1}_{RSU_2}}),$
$split(sp\_evidence_1),\ split(sp\_evidence_2),$
$decrypt - asymm(\{ECN\}_{K_{TS}}, K^{-1}_{TS}),$
$generate\_ticket(ECN, t_1, t_2, t_{limit}),$
$\textbf{\textit{Forget}}(\{sp\_evidence_1\},$
$\textbf{\textit{Forget}}(\{sp\_evidence_2\},$
$\textbf{\textit{Forget}}(\{ECN\}_{K_{TS}}),$
$\textbf{\textit{Forget\_secret}}(\{ECN\})\}$

$bvr_3^{TS} = \{Send(TA, m_7); \quad - \ \}$

Note that, although the value $sp\_ticket = \{TS \parallel ECN \parallel t_1 \parallel t_2\}_{K_{TA}}$ is not forgotten, the content can only be retrieved by TA.

Expected security and privacy goals: The following list enumerates the goals in relation to vehicular identity privacy.

G1: $POSS(TS)\_Final = POSS(TS)\_Initial \cup \{sp\_ticket\}$. POSS(TS) has only increased with $\{sp\_ticket\}$.

G2: $Observers(ECN) = \{OBU\}$. Secret ECN must only be known by OBU.

Formal proof: Only those steps in relation to the satisfaction of goals are shown.

$bvr_2^{TS} = \{Receive(RSU_2, m_6);$
$\quad POSS(TS) = POSS(TS) \cup \{sp\_evidence_1\}_{K^{-1}_{RSU_1}}\}$

$Decrypt - signature(m_3, K_{RSU_1}),$
$\quad POSS(TS) = POSS(TS) \cup \{sp\_evidence_1\}$

$\textbf{\textit{Forget}}(\{sp\_evidence_1\}_{K^{-1}_{RSU_1}}),$
$\quad POSS(TS) = POSS(TS) - \{sp\_evidence_1\}_{K^{-1}_{RSU_1}}\}$

$Decrypt - signature(m_6, K_{RSU_2}),$
$\quad POSS(TS) = POSS(TS) \cup \{sp\_evidence_2\}$

$\textbf{\textit{Forget}}(\{sp\_evidence_2\}_{K^{-1}_{RSU_2}}),$
$\quad POSS(TS) = POSS(TS) - \{sp\_evidence_2\}_{K^{-1}_{RSU_2}}\}$

$split(sp\_evidence_1),$
$\quad POSS(TS) = POSS(TS) \cup \{\{ECN\}_{K_{TS}}, t_1\}$
$\quad$ Note that the *Update* function renders the following result:
$\quad Observers(ECN) = Observers(ECN) \cup \{TS\}$

$split(sp\_evidence_2)$,
$POSS(TS) = POSS(TS) \cup \{\{ECN\}_{K_{TS}}, t_2\}$
$generate\_ticket(ECN, t_1, t_2, t_{limit})$,
$POSS(TS) = POSS(TS) \cup \{TS \parallel ECN \parallel t_1 \parallel t_2\}$
Forget$(\{sp\_evidence_1\})$,
$POSS(TS) = POSS(TS) - \{sp\_evidence_1\}$
Forget$(\{sp\_evidence_2\})$,
$POSS(TS) = POSS(TS) - \{sp\_evidence_2\}$
Forget$(\{ECN\}_{K_{TS}})$,
$POSS(TS) = POSS(TS) - \{ECN\}_{K_{TS}}$
Forget_secret$(\{ECN\})$,
$POSS(TS) = POSS(TS) - \{ECN\}$, **(G1)**
$Observers(ECN) = Observers(ECN) - \{TS\}$, **(G2)**

□

5 Conclusions

In this paper we have described what we believe is tractable and powerful formalism for the formal verification of privacy–enhancing cryptography protocols. The original logic introduced by Rubin offered a major contribution for the reasoning of non–monotonic protocols although it needed a few amendments and updates to it, for example, the concept of signature verification used in Rubin's logic was, in our opinion, invalid. The new extended logic L–PEP has served to formally verify the dual signature of the protocol SET and a speed ticketing electronic *non–monotonic* system. The logic has enabled us to reason about private data protection laws by which transportation authorities can only store information about offender vehicles and where tracing of non–offenders is not permitted.

Future Work. Currently, we are analyzing a privacy–enhancing authentication and access control protocol based on blind signatures and recursive hashing.

Additionally, a more advanced version of the speed ticketing electronic service here described, making use of identity based encryption, is under verification using the described formalism.

Finally, the extended logic also intends to set the basis for future automated design of multi-party privacy–enhancing cryptographic protocols by means of evolutionary computation and artificial intelligence techniques. Previous works on this area are [13]. The logic just presented will allow for similar techniques to synthesize privacy related protocols.

References

1. Meadows, C.: Formal methods for cryptographic protocol analysis: emerging issues and trends. IEEE Journal on Selected Areas in Communications 21(1) (2003)
2. Privacy and identity management for europe (prime). Privacy and Identity Management for Europe (PRIME), https://www.prime-project.eu

3. Burrows, M., Abadi, M., Needham, R.: A logic of authentication. ACM Transactions on Computer Systems 8(1), 18–36 (1990)
4. Abadi, M., Tuttle, M.: A semantics for a logic of authentication. In: Proceedings of the ACM Symposium of Principles of Distributed Computing, pp. 201–216. ACM Press, New York (1991)
5. Moser, L.: A logic of knowledge and belief for reasoning about computer security. In: Proceedings of the Computer Security Foundations Workshop II, pp. 57–63 (June 1989)
6. Rubin, A.D.: Nonmonotonic cryptographic protocols. In: Proceedings of the Computer Security Foundations Workshop, pp. 100–116 (1994)
7. Meadows, C., Syverson, P.F.: A Formal Specification of Requirements for Payment Transactions in the SET Protocol. In: Hirschfeld, R. (ed.) FC 1998. LNCS, vol. 1465, pp. 122–140. Springer, Heidelberg (1998)
8. Xu, Y., Xie, X.: Analysis of electronic commerce protocols based on extended rubin logic. In: Proceedings of the 9th International Conference for Young Computer Scientists, Washington, DC, USA, pp. 2079–2084. IEEE Computer Society, Los Alamitos (2008)
9. Xu, Y., Xie, X.: Analysis of authentication protocols based on rubin logic. In: 4th International Conference on Wireless Communications, Networking and Mobile Computing, WiCOM 2008, pp. 1–5 (2008)
10. Xu, Y., Xie, X.: Security analysis of routing protocol for manet based on extended rubin logic. In: IEEE International Conference on Networking, Sensing and Control, ICNSC 2008, pp. 1326–1331 (2008)
11. Teepe, W.: On ban logic and hash functions or: how an unjustified inference rule causes problems. Autonomous Agents and Multi-Agent Systems 19(1), 76–88 (2009)
12. Agray, N., van der Hoek, W., de Vink, E.P.: On ban logics for industrial security protocols. In: Dunin-Keplicz, B., Nawarecki, E. (eds.) CEEMAS 2001. LNCS (LNAI), vol. 2296, pp. 29–36. Springer, Heidelberg (2002)
13. Alcaide, A., Estévez-Tapiador, J., Hernandez Castro, J., Ribagorda, A.: Nature–inspired synthesis of rational protocols. In: Rudolph, G., Jansen, T., Lucas, S., Poloni, C., Beume, N. (eds.) PPSN 2008. LNCS, vol. 5199, pp. 981–990. Springer, Heidelberg (2008)

Surveillance, Privacy and the Law of Requisite Variety

Vasilios Katos[1], Frank Stowell[2], and Peter Bednar[2,3]

[1] Department of Electrical and Computer Engineering
Democritus University of Thrace, Greece
[2] Information Systems Research Group School of Computing
University of Portsmouth, UK
[3] Department of Informatics, Lund University, Sweden
vkatos@ee.duth.gr, frank.stowell@port.ac.uk, peter.bednar@ics.lu.se

Abstract. In both the academic literature and in the media there have been concerns expressed about the level of surveillance technologies used to facilitate security and its effect upon privacy. Government policies in the USA and the UK are continuing to increase surveillance technologies to counteract perceived terrorist threats. Reflecting upon Ashby's Law of Requisite Variety, the authors conclude that these policies will not meet espoused ends and investigate an alternative strategy for policy making. The authors develop a methodology by drawing on an isomorphy of concepts from the discipline of Macroeconomics. This proposal is achieved by considering security and privacy as economic goods, where surveillance is seen as security technologies serving ID management and privacy is considered as being supported by ID assurance solutions. As the means of exploring the relationship between surveillance and privacy in terms of the proposed methodology, the authors use scenarios from a public report commissioned by the UK Government. The result of this exercise suggests that the proposed methodology could be a valuable tool for decision making at a strategic and aggregate level.

Keywords: Ashby's Law of Requisite Variety, Systems, Macroeconomics, Surveillance; Security, Privacy.

1 Introduction

In this paper we raise questions about the effectiveness of what appear to be policies regarding the installation of surveillance technologies. We ask if such technologies are eroding individual privacy. In this paper we investigate whether it is possible to strike a balance between surveillance as a means of citizen protection and at the same time preserve privacy. In the context of the paper the term "surveillance technologies" is used to describe all monitoring technologies that continuously capture data on a wide scale.

If we assume that the purpose of surveillance is to exercise control over a situation it monitors e.g. crime reduction, terrorist activities, then we must take into consideration both the subject and the means of monitoring that subject.

J. Garcia-Alfaro et al. (Eds.): DPM 2010 and SETOP 2010, LNCS 6514, pp. 123–139, 2011.

To this end we reflect upon Ashby's [1] argument that in any system "...if a certain quality of disturbance is prevented by a regulator from reaching some essential variables then that regulator must be capable of exerting at least that quantity of selection" (pp. 229). In other words the variety in the controller must have as much variety as that which it seeks to control. This revelation became known as Ashby's Law of Requisite Variety (LRV) which explains that in order to bring about effective control over any system the control mechanism must be capable of addressing as many different outcomes that it is possible for a system to develop. The more complex the system the more difficult it is to predict its behaviour and the more difficult to exercise control.

It is axiomatic that in order to manage complexity we need to reduce its variety and by thinking in terms of, "...heuristics rather than algorithms" which, as Beer [2] reminds us that it is, "...at once a way of coping with proliferating variety...instead of trying to organise it in full detail, you organise it only somewhat; you then ride the dynamics of the system" (p.53).

The question we raise is what impact are the measures ostensibly being set up to strengthen security having upon each citizens right to privacy? To this end this paper is an account of an attempt to develop a decision support tool to assist in strategic and aggregate decision-making relating to surveillance. The LRV suggests to us that such policies cannot achieve the desired end of full protection and we argue that continuance of such policies will erode individual privacy. As a consequence we reflect upon alternative strategies and describe a methodology that we believe to be useful to decision makers. The paper includes outcomes from empirical studies in which attitudes and concerns about the use of personal data were revealed which provide support for the contention that there is a growing unease about the use of technological surveillance.

2 Surveillance and Modelling

Modelling the relationship between surveillance, security and privacy means that certain assumptions have to be made as any model of such a situation is a simplification. Because of the unlimited variety in the system it is impossible to give a full specification and impossible to produce an exact algorithm. As Beer [2] points out "The strange thing is we tend to live our lives by heuristics and try to control them by algorithms" (p.53). But in adopting an heuristic it provides a route to "the eventual discovery of a strategy" [2] (p.55). We cannot know the future and as a consequence cannot work out a strategy in advance but what we should do instead is to devise a system of control that enables us to learn and modify its unwanted effects. Beer's point made in another context, is that our concern is to link a high variety input with a high variety output [2] and as a consequence we have to find ways in which we can reduce the variety of the target system or increase the variety of the control subsystem. We suggest it is possible to produce a description of the general direction in which a given

subject is likely to move and to provide a means of assessing the effect of the actions taken. However, the complexity of the situation means that modelling it is not a straight forward application of a formula.

2.1 Surveillance and Privacy

The challenges in understanding and modelling privacy are primarily two-fold. The first challenge relates to developing a definition of privacy; an exercise which is difficult because of the "incompatible" and rich characteristics of privacy. Brunk [5] said that "[P]rivacy is a matter of intellectual and philosophical thought and retains few tangible characteristics, making it resistant to simple explanation". As such, finding an objective and universally accepted definition of privacy could be argued to be a futile exercise [6]; in fact Odlyzko [7] goes further by speculating that the privacy problem is intractable.

The lack of a clear definition of privacy within the literature means that it becomes necessary to adopt one which has resonance with the underpinning thinking within this research, namely that there should be a transparency about the level and type of surveillance used. In this sense we subscribe to Westin's definition [8] summarised as the ability of an individual to control the terms under which personal information is acquired and used. Westin's definition is appealing because it associates privacy with the user, rather than the environment. That is it relates to the user's opportunities and capabilities to protect the outflow of their personal information and private data.

The second point we make is that privacy seems to be interwoven with security in a way that any study or treatment of privacy in isolation can result in misleading conclusions. For example, when developing security oriented systems most member states of the European Union are careful not to ignore citizens privacy (see for example the E.C. Directive 1997/66/EC of the European Parliament). In contrast in the United States, the Total Information Awareness (TIA) project, which focussed on applying surveillance and information technology to track and monitor "terrorists" was terminated by the Congress [4]. Following the terrorists attacks on the USA and EU the United Kingdom proposed the European passenger name record framework be expanded to increase the powers of data collection and use outside the air travel remit as a means of combating organised and serious crime.

Clearly this is a complex problem with issues of security and personal freedoms interwoven in such a way that increasing one aspect may have a detrimental effect upon the other. Whilst accepting and the concerns of the UK Government about terrorist attacks on UK citizens and acknowledging the warning from the House of Lords [9], we believe that it is important to provide a transparent system of control such that citizens are aware of what is happening and be assured that they are not over monitored. We believe that most citizens intuitively feel a symbiotic relationship between security and privacy such that the presence of one justifies the survival of the other. Our position, with specific caveats, is that;

– Privacy is dependent on security, as it cannot be claimed that privacy is achieved if there are no security mechanisms in place to protect it. Cryptography is an example of protecting confidentiality and of the dependence of security protecting privacy.
– Security is dependent on privacy with respect to demand; if there is no demand for privacy, some security controls will be deemed redundant and unnecessary. For example, if there is no requirement for anonymity in web browsing, then there will be no demand for anonymous proxies; or if the personal data are not required to be private then there would be no data confidentiality on the underlying data types.

2.2 Modelling the System

Privacy is a human, socially driven characteristic and as such research into privacy metrics and calculi inherently involves human attributes such as perceptions and beliefs, demographic dependent data [3]. Our view is that the application of a micro- or macro-economic ideas to study on privacy and security can only be meaningful if the sociological context is taken into consideration. One such interdisciplinary type of research by applying social network theory to study the economics of privacy as influenced by mass surveillance is performed in [10].

Reductionist modelling methods of such complexity as the relationship between privacy and security is problematic as it is unquantifiable. But within the context of an economic analysis, avoidance of such deficiencies that are inherent in reductionism could be achieved by moving from micro to macroeconomics. Typically by adopting modelling methods from micro or macroeconomics we are able to produce a model which is useful because it allows us to develop models where not all variables are known. The relationship between privacy and security is one such situation. Furthermore macroeconomics seem more appropriate than microeconomics since the system under consideration has a wide scope (the participants or users are members of a society) requiring an investigation on an aggregate level.

3 ID Management versus ID Assurance

Increasing numbers of individuals using Internet services has resulted in a paradigm shift from an emphasis on business use to citizen and e-governance. In [11] Crosby points out the paradigm shift that has brought privacy obligations to the foreground and disturbed the security agenda by requiring analysts to consider possibilities for migrating from ID management to ID assurance solutions.

Although ID management and ID assurance have common grounds they are two sides of the same coin. ID management depends on controls where both security and ID management support surveillance. ID assurance depends on security where both ID assurance and security support privacy.

The ID management versus ID assurance contrasts are reflected and analysed. In a project commissioned by the UK Government [11] Crosby explored how

an ID Management solution could be developed and used by private and public sectors in the UK which would be of benefit to them and citizens. The study was carried out involving over 100 organisations with and individuals being consulted from all sectors and sizes of companies. We have adopted [11] as a main point of reference in this research for the following reasons:

- it was conducted at a level appropriate to the research requirements of this paper (i.e. on an aggregate or macro level);
- it acknowledged the wider international context and as such the findings can have a wider remit;
- the outcome of the study identified the purpose and context of the two schemes (ID management and assurance) in relation to the needs of surveillance and data collection.

The researchers had no control over the study and its conclusions and its contents are open for independent analysis.

3.1 Methodology

Both Microeconomic and Macroeconomic theory are accustomed to handling ill-defined problems. For example, consider the functions of supply and demand which are used to define a product or service market. The underlying market can be viewed as an open or general system in which not all relations and variables can be accounted. Macroeconomic theory allows the introduction of assumptions that transfer the problem to a partial system. In economics this is the so-called ceteris paribus assumption, or all other things being equal, under which levels of specific variables are considered fixed or constant [12] (Schlicht, 1985). Such analysis will provide a snapshot of the problem so in order to give due consideration to the decision making process it is necessary to have means of relaxing the assumption in a controlled way to allow the study of certain scenarios. Relaxing is done when we introduce assumptions in the model which affect some of the controlled variables in "predictable" or generally acceptable ways.

We will now extend our argument to see what effect this might have on the relationship between privacy and security technologies. We present the two alternatives offered by Crosby which are compared to the current situation, a mosaic of different systems containing a plethora of personal data, as elaborated below. This is followed by a systematic application of the proposed methodology which invokes an isomorphy of macroeconomic theories and tools.

Baseline case: "Many eggs in many baskets". There is no national ID card scheme in the UK. There are many strategies and solutions for both public and private sector for individuals to verify their identity, depending on the relevant requirements of the underlying industry (see [11]). For example, in banking opening a new account requires proof of identity which can take the form of a driving licence (or a passport) and a utility bill to the individuals' home address, this process even applies to existing and new customers that wish to open a new savings account. Telephone or on-line banking transactions require the

customer to provide a series of answers to obscure questions such as mothers maiden name or shoe size. To the average citizen such checks can be tedious but to the criminal of average intelligence falsifying an identity which would satisfy this level of enquiry is relatively straight forward. The Information Technology Association of America [13] reported a growing level of identity theft and fraud and attributed this increase to the U.S. system which makes it easy for criminals to obtain false identities. The report points out the ease at which it is possible to gain an identity providing a startling example that "Identity thieves need not always go to the trouble of trying to craft their own phony identities. Sometimes they can simply purchase the documents by bribing department of motor vehicle personnel" [13].

In the public sector once a (temporary) national insurance number has been obtained then an individual can begin to access a variety of state benefits and also have an identity that will satisfy most other institutions. In each of these instances records will be created for individuals who are ostensibly protected by the UK 1998 Data Protection Act which forbids the exchange of data between separate agencies.

The above situation results in the creation of disparate systems where some hold incomplete and potentially conflicting data as well as overlapping and redundant entries. Such a situation will produce an assortment and a multitude of ID management and ID assurance systems. Not surprisingly an attempt to develop interfaces between all these different systems is a non-trivial task. For example, in order to operate many public sector agencies frequently circumvent their lack of interface by inventing "simple" yet risky procedures such as the "Frisbee Network" of CD exchanges [14],[15].

ID Management: "Many eggs in one basket". The principle behind ID management is to increase the amount of information collected for every citizen [11]. Information is held for the purpose of protecting society from fraudulent behaviour, criminal activities, and terrorism [16]. ID management is about the unreserved collection of data, which includes identification as well as behavioural data. This will have an impact to security as it was demonstrated empirically in [17] that active monitoring of an information system is indeed a deterrent to computer abuse. The practical implication of this principle requires surveillance technologies to be implemented and coordinated and results in a trade off between security and privacy. The result is an increase in the level of complexity and cost because by storing a plethora of personal data much will fall into special legislative categories, high confidentiality. Ways of manipulating, handling, processing and combining the data will also add to the complexity of its management. For a complex system to be manageable it is necessary that the flexibility of the system and controls will be reduced [18]. We are reminded that "...every real machine embodies no less than an infinite number of variables" ([1], p.15).

Potentially ID management creates a single point of failure, because if data mining is applied the data could be accessible by unintended parties or authorities. Inference control is an issue in many dimensional databases (see for example [19]) because if the data are available then they can be used for other than the

intended purpose of managed citizen identification. Security technologies focus on meeting the goals of prevention, protection and recovery [21] but the latter two do not contribute to the goal of preserving privacy. Privacy is preserved when information about a person is only disclosed to a specific group of entities (or subjects), and is one failure which results in unauthorised disclosure of information. When dealing with a centralised repository of data the problem of unauthorised disclosure becomes even more challenging than providing a set of access control mechanisms because of the data inference problem, i.e. data leaking information about other, higher classified data. Challenges to privacy and security in this case is represented as an inference control problem [20].

ID Assurance: "An egg per basket". The fundamental difference between an ID management and an ID assurance system is that the latter intends to support the development of a trusted system for identity verification. This is achieved through minimizing the amount of data held on each citizen, thus supporting privacy. The data itself is of low confidentiality as the data relate only to identification and not behaviour. As a direct consequence surveillance is not required and privacy is supported. Because of the relative limited scope, amount and confidentiality of the data, security requirements will be less complex and less expensive.

4 Applying the Methodology

First, we consider those security controls that aim to protect privacy. In this context we use the term security technologies in a generalised fashion to encompass all types of security controls, both technical and organisational. Accordingly we adopt an aggregate view by splitting the security technologies into two markets namely, the defensive security market and the adversarial security technologies market. The defensive security market consists of all those technologies which are used to protect privacy in a defensive manner. These are typically access control techniques involving services such as identification, authentication, authorisation and accountability. From a practical perspective these services can be implemented by defensive security technologies such as cryptography, firewalls, intrusion detection systems, anti-virus software and biometrics. Adversarial technologies consist of all those technologies which are used proactively to audit the defensive security controls. Examples of adversarial technologies include security scanners, vulnerability and penetration testing tools.

The distinction we make between the two categories relaters to purpose and intention. For the purpose of this research we assume that defensive technologies are used solely for benign purposes, but adversarial technologies can be used for both benign or malicious purposes. An example of a benign use of adversarial technologies is ethical hacking which is normally found within the kernel of a security assessment exercise. We emphasise that both defensive and adversarial security technologies are required for protecting privacy. For example, removing ethical hacking would render privacy protection mechanisms incomplete and eventually ineffective.

4.1 Initial Case: An Aggregate View

Advancing the argument in [22] we consider the current situation of many types of user identification systems with varying degrees of interfacing designed to meet a range of security requirements some of which may be conflicting or incompatible. The data are dispersed throughout these user identification and authentication systems. As such pieces of data may be overlapping, redundant, incomplete and conflicting. Within this mosaic of data and security requirements we identify pairs of relationships between certain variables. Specifically, the following assumptions are made for the two markets [22]:

A. Security technologies market:

- Security Demand versus Price. Security technologies are seen as economic goods. The aggregate demand (SD) of these technologies depends on their price (P). The demand for security technologies is represented by a monotonically decreasing function $SD = f(P)$ i.e. the lower the price of security technologies (P), the higher the demand of security technologies (SD) creating an inverse relationship between quantity demanded for security technologies and price.
- Security Supply versus Privacy. The new economic analysis of privacy offers arguments to those who believe that individuals make an effort to act rationally [23] and choose whether to adopt privacy technologies. We assume that security is required in order to have privacy, because privacy cannot exist without security. In this case the supply for security technologies function $SS = g(V)$, says the more important (higher) privacy (V) the greater the quantity of security technologies (SS) i.e. there is a positive relationship between supply of technologies and privacy. It is important to note the form and shape of security here because privacy is bounded by the supply of the security technologies, but security also depends on privacy such that if privacy disappeared, the portion of security technologies dealing with enhancing privacy would also disappear. In other words we argue that there is a symbiotic of relationship between security and privacy with some advantage allocated over security. Removal of privacy would not necessarily result in a collapse of security, but it is likely that the latter would exist in a different form (e.g. a totalitarian or dictatorial regime).
- Security supply versus security demand. Branson and Litvack [24] and Dornbush and Fischer [25] suggest that market forces and economic laws, if left alone, will eventually push (security technologies) demand to equilibrium with (security technologies) supply, regardless of their initial allocation. This is represented by the identity function $SD = SS$, or $f(P) = g(V)$.

The model in Fig.1 which represents the security technologies market is constructed by chaining the three pairs of relationships presented above in a way that an equilibrium or relationship between price and privacy can be obtained in $Q1$

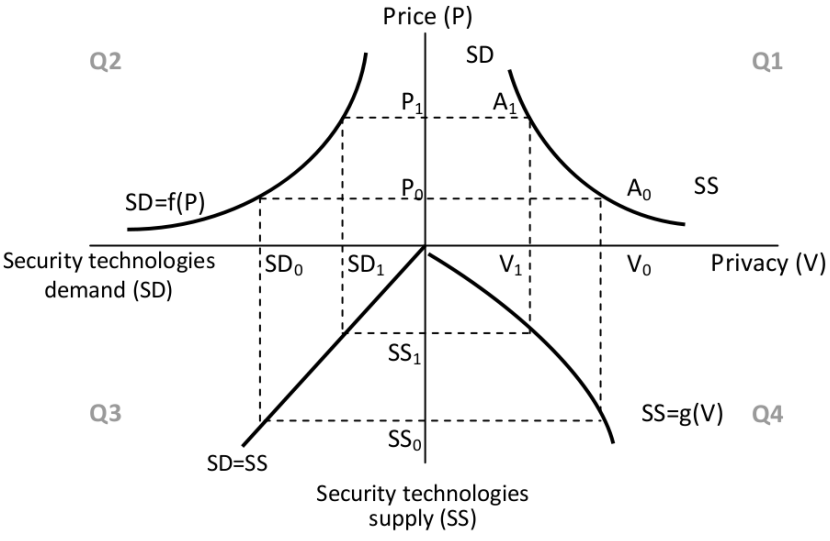


Fig. 1. Security technologies market

B. Adversarial technologies market: A fundamental distinction between security and adversarial technologies, is that in the latter there needs to be a distinction based on the intention or purpose of the technology i.e. is it benign (for protecting privacy), or malicious (for defeating privacy)? In a manner similar to the security technologies market above we arrive at the relationships as shown in Fig. 2 where the positively sloped $SM - SB$ curve in $Q1$, describes equilibrium pairs of P and V, representing the adversarial technologies market (for a more elaborate description of the approach refer to [22]).

So far we have derived two pieces of geometric equipment. One gives the equilibrium pairs of P and V in Fig.1, i.e. the $SD - SS$ curve in the security technologies market and the other gives the equilibrium pairs of P and V in Fig.2, i.e. the $SM - SB$ curve in the technologies of the adversary market. By placing these two curves in the same quadrant which is achieved by solving the two equilibrium equations $f(P) = g(V)$ and $SS^* = h(P) + k(V)$ simultaneously, we can find the single P, V pair that gives equilibrium in both markets point $E(P_E, V_E)$ of the intersection of the $SD - SS$ and $SM - SB$ curves (typically this is done by superimposing the two $Q1$s from the two figures).

Summary. At this stage the above exercise is incomplete for two reasons. First, as with most qualitative variables there is no acceptable metric for privacy. Privacy being equal to V_E has limited meaning or substance as there is no objective metric for privacy. Yet it is implicitly accepted by developing this approach that privacy could be captured by an ordinal variable agreeing with the wider consensus that certain actions may reduce or increase privacy, despite

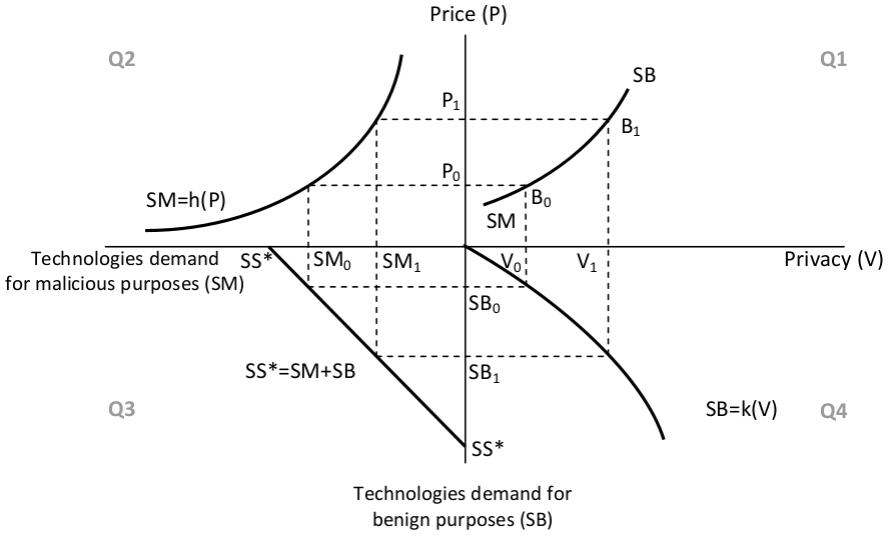


Fig. 2. Adversarial technologies market

not being able to quantify the precise changes. An indicative example revealing the ordinal scale of privacy is the true statement: "Privacy will decrease if ISPs are legally allowed to monitor their customers' internet browsing actions and share this information with any third party". Second, the assumption of "ceteris paribus" means that the model exhibits a snapshot of a market or environment so we are transforming a general to a partial system by making the levels of all other variables constant.

Nevertheless, the added value of this model is in its ability to allow the ceteris paribus assumption to be relaxed by accommodating changes to existing or introducing new assumptions. By doing this we can run what-if scenarios and track the relative changes of the qualitative variables such as privacy. The output of the model is the recommendation of hypotheses that will create further empirical research activities. The validation of the model was undertaken through the use of a publically accessible scenario chosen because its known outcome provides a point of reference and a pilot study.

At this point it should be clarified that this initial case described above has not taken into consideration bounded rationality and the behavioural paradoxes that some researchers have established (see for example the survey conducted in [26] it was observed that the users consent to the commercial use of their personal data for petty amounts of cash (eg. 1) but are not willing to spend similar amounts for the protection of these pieces of data). Behavioural paradoxes can be incorporated in the model by empirically estimating the underlying functions and more specifically their slopes. Such exercise is outside the scope of this paper but it is a vital area for developing this approach further.

Comparative statics exercise: ID Management. In ID management an abundance of personal data is collected centrally which is controlled by a limited and select group of individuals or organisations. Comparing this policy to the base case, we can make the following observations:

M1. As all data will reside in one system, the complexity of the system will increase. The larger data schemas will result in a larger number of permutations of states, data and interfaces accessing the system. Reflecting upon Ashby's LRV shows this will result in a larger number of controls. In order to maintain the same amount of privacy as before, more security will be required.
M2. With "all eggs in one basket", the data type with the highest privacy requirements is likely to set the security requirements of the overall ID management system.
M3. Increased complexity of the ID management system means security will become more expensive and, assuming a fixed budget, aggregate demand will appear to drop.
M4. Some security technologies will be available to a limited, select group.

Assumptions M1 and M2 would cause a leftward rotation of the privacy-security supply curve in $Q4$, as shown in Fig. 3. Assumptions M3 and M4 produce a rightward shift of the demand curve, shown in $Q2$. The movement of the curves in $Q2$ and $Q4$ would have a knock-on effect to the relationship between privacy and price in $Q1$, by causing a leftward shift. The new equilibrium function is represented in $Q1$ in Fig. 3 by the dashed line $SD' - SS'$. The new position of the

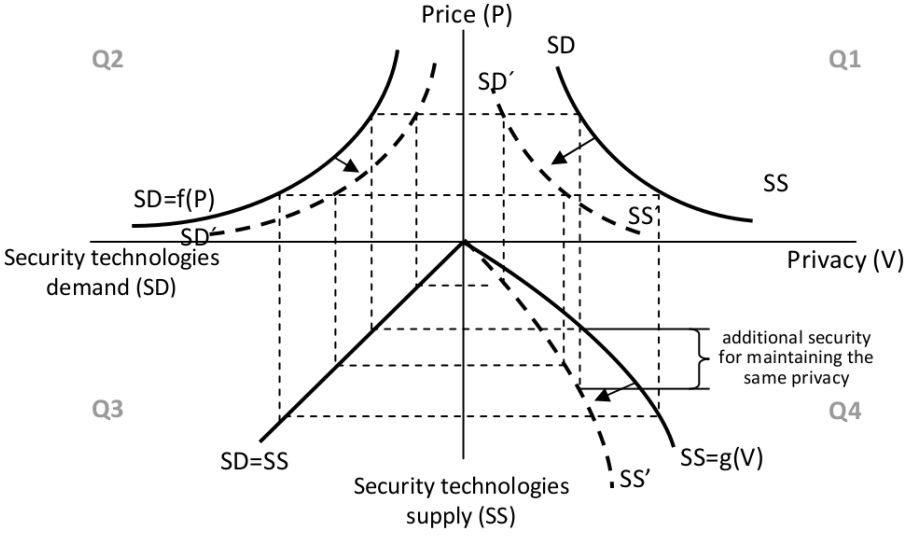


Fig. 3. The impact of the introduction of a central ID management system to the security technologies market

equilibrium function is definitely to the left of the initial position, because the effects of the changes in the security technologies demand and supply functions are reinforcing each other.

Accepting the predicted outcome from the model we expect an ID management solution is expected to affect the adversarial market as follows:

M5. In line with M1, additional auditing or use of adversarial technologies for benign purposes would be required in order to maintain privacy to a certain fixed level.

M6. The value of the equivalent database holding the vast amounts of personal biometric and biographical data will be substantially higher than any smaller system containing a subset of the data. Therefore the demand for adversarial technologies for malicious use would increase, as the target system would be appealing to attackers.

Assumption M5 would influence accordingly by a leftward rotation of the privacy benign adversarial technologies relation shown in $Q4$. Assumption M6 would cause a leftward shift of the demand curve in $Q2$, since for a fixed price there will be a higher demand of malicious use of adversarial technologies due to the higher potential gain of compromising the system. These assumptions are reflected in Fig. 4, where the security technologies curves are superimposed in $Q1$ to show the new market equilibrium E'.

Comparing points E and E' (comparative statics) it is seen that privacy decreases from V_E to $V_{E'}$, whereas the price seems to remain relatively stable. Intuitively, this could be explained by arguing that any attempt to reduce the

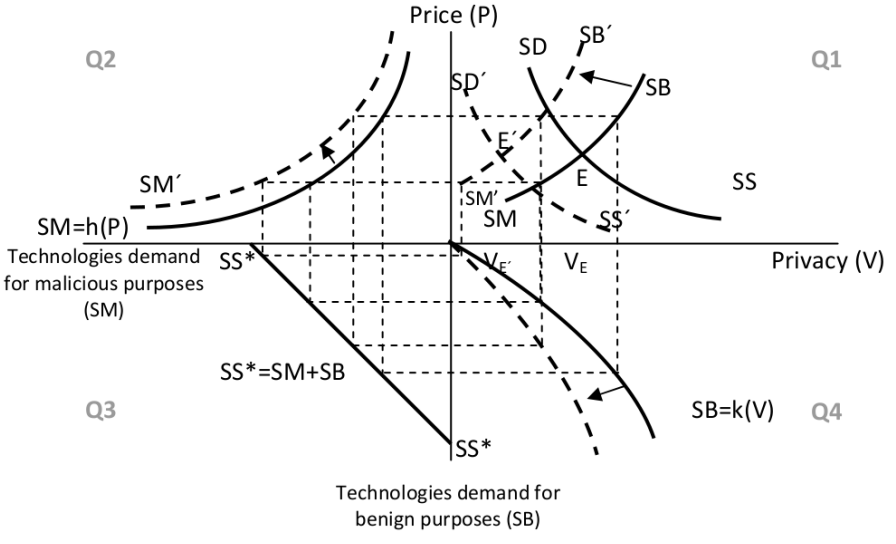


Fig. 4. The impact of the introduction of a central ID management system to the adversarial technologies market

price by trading off privacy is "compensated" by a response - increase - in adversarial systems used for malicious purposes. At an aggregate level we observe a drop in privacy with no benefit to price.

Comparative statics exercise: ID Assurance. The ID assurance scenario shows a minimum amount of data stored in the ID verification system, but also that this data would be of a low value in terms of confidentiality and usability. We could view the system as a public directory of high integrity requirements. In an ID assurance system, the following assumptions may be drawn:

A1. Overall, there are significantly less data than the other two user identification systems presented earlier, particularly due to the fact that no behavioural data would reside in the system.
A2. The citizens are seen as active and effective parts of the security processes largely through the high frequency of use of the system [11].
A3. The overall complexity of the system will be reduced, as the confidentiality requirements would be low.

Assumptions A1 to A3 would effectively cause the security technologies market to exhibit a higher amount of privacy for a certain level of supply of security technologies. That is, given a certain supply level of security technologies, privacy is expected to be higher in this scenario, since the volume and confidentiality of the data will be less than in the base case. This would result in a rightward rotation of the privacy versus security curve in $Q4$ in Fig. 5. Moreover, assumption A3

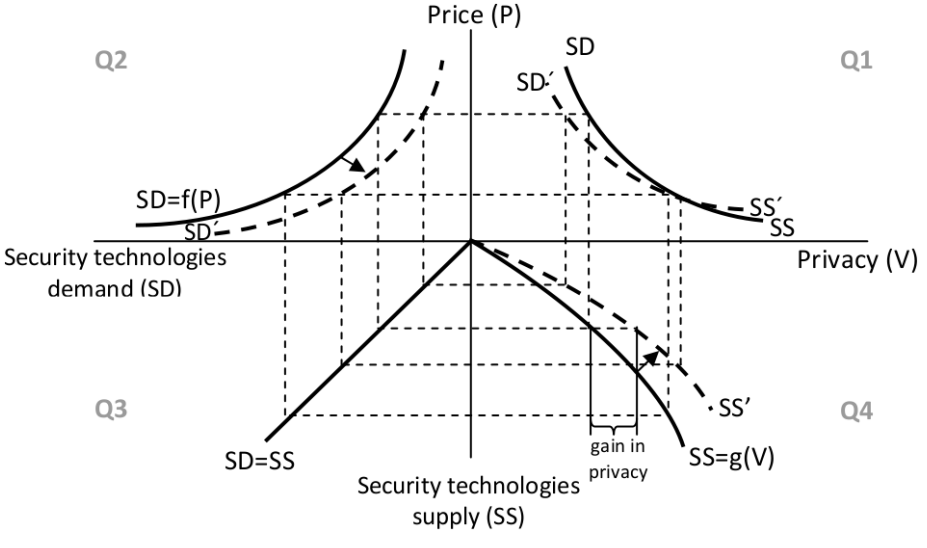


Fig. 5. The impact of the transition to an ID assurance system to the security technologies market

would result to a rightward shift of the security demand versus price curve in $Q2$ in Fig. 5, since the aggregate demand for security technologies is expected to drop (there will be fewer data and less complex systems to protect).

From the resulting shift of $SD - SS$ curve in $Q1$ it can be seen that there is no significant or clear-cut movement of the privacy versus price curve. The decrease in demand of the security technologies ($Q2$) compensates any clear gain in privacy ($Q4$). With respect to the adversarial technologies market, the following assumptions are made:

A4. In line with A3, a lower complexity would in turn require smaller security auditing efforts in order to maintain a certain level of privacy.
A5. The ID assurance system would be of a lower value to attackers and a less attractive target. As such, we argue, that the demand of adversarial systems for malicious purposes would drop.

Assumption A4 would cause a rightward rotation of the privacy versus benign adversarial technologies curve in $Q4$ in Fig. 6. Assumption A5 would be reflected in the model as a rightward shift of the demand curve in $Q2$. The consequences of the combined shifts of the two curves to the privacy versus price curve in $Q1$ are shown as a distinct rightward shift. The security technologies curves are superimposed in $Q1$ to show the new market equilibrium E'.

Comparing points E and E' it can be seen that there is a clear increase in privacy and a decrease in price. Intuitively this outcome seems correct, since there will be less security systems required to protect reduced data sets for a

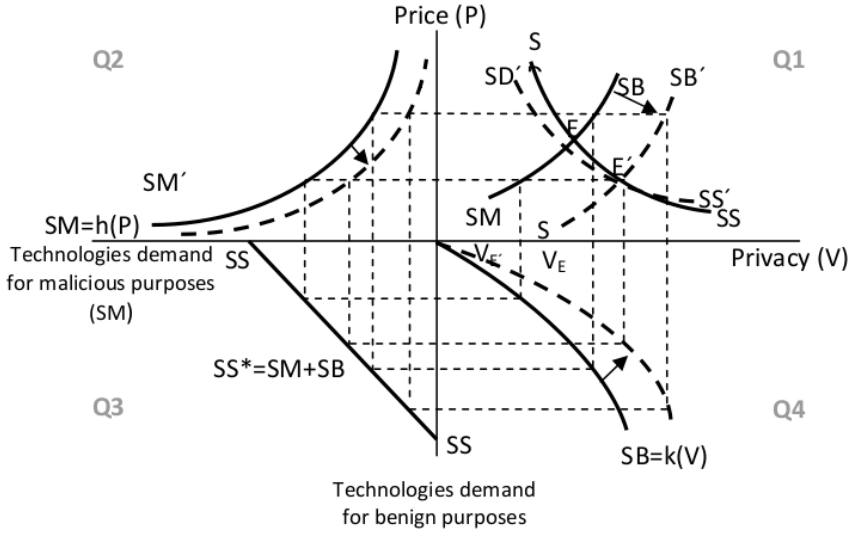


Fig. 6. The impact of the transition to an ID assurance system to the adversarial technologies market

specific purpose (identity verification) and consequently of a limited value to an attacker. If the data are mainly unclassified or have low confidentiality requirements, privacy threats would be out of context. This is due to the decoupling of the ID assurance system from confidentiality needs. However the integrity requirements would remain high, as unauthorised modification of the relatively few user identification data types would be a valid threat for the identification system.

5 Conclusions

In this paper the relationship between security and privacy was considered from an aggregate perspective. By observing the shift from monitoring to surveillance and from the business employee to the citizen (now an end user of e-Government) we have embraced a key idea from General Systems Theory namely the belief in the possibility of an "isomorphy of concepts, laws and models in various fields" ([27], pp. 35). To this end we incorporated the ideas from the cross methodology from macroeconomics as a means of investigating the relationships between privacy security based upon cost and consumption. Specifically we considered security and privacy to be economic goods and as a consequence price functions were attached to them.

Taking into account Ashby's Law of Requisite Variety we argue that excessive surveillance will not solve security issues as the level of threat and intensity is not constant and was unpredictable. Two alternative scenarios or "futures", namely ID management and ID assurance were considered using scenarios taken from a publically available report. This report had been commissioned by the UK Government and provides the reader with a rich source of research. Using the model our analysis showed that ID management, characterised by the unreserved and centralised collection of citizen's data, offers no perceived benefits to security, but the level of privacy will decrease with no significant price trade-off. In other words there will be less privacy for the same price and as a consequence privacy would be relatively more expensive. In security terms we do not expect improvement either, as a centralised repository of personal data will raise the complexity of the user identification system and the security controls and increase its attractiveness to malicious adversaries because of its added value.

In the case of ID assurance we have an identity verification system with a minimal amount of data exhibiting low confidentiality allowing participation of the end users stakeholders through high frequency of use. Such an identification system will have low complexity. Our analysis showed that privacy will increase at a lesser price. This is not surprising because the ID Assurance system would not only be of low value to attackers, but also the security employed would serve more privacy and less surveillance since user identification would depend on the integrity rather than the confidentiality of unclassified data.

References

1. Ashby, W.R.: Design for a Brain. Halstead Press, New York (1960)
2. Beer, S.: Brain of the Firm, 2nd edn. Wiley, Chichester (1981)
3. Westin, A.F.: The 1996 Equifax-Harris Consumer Privacy Survey. Equifax Inc., Atlanta (1996)
4. Williams, M.: The Total Information Awareness Project Lives On (2006), http://www.technologyreview.com/Infotech/16741/
5. Brunk, B.: Understanding the Privacy Space. First Monday 7(10) (2002), http://www.firstmonday.org/Issues/issue7_10/brunk/
6. Klopfer, P., Rubenstein, D.: The Concept Privacy and its Biological Basis. Journal of Social Issues 33, 22–41 (1977)
7. Odlyzko, A.: Privacy, Economics, and Price Discrimination on the Internet. In: ACM, Fifth International Conference on Electronic Commerce, pp. 355–366 (2003)
8. Westin, A.: Privacy and Freedom. Atheneum, New York (1967)
9. House of Lords, European Union Committee.: The Passenger Name Record (PNR) Framework Decision, 15th Report of Sessions 2007-08, London, the Stationary Office Limited (2008)
10. Danezis, G., Wittneben, B.: The Economics of Mass Surveillance and the Questionable Value of Anonymous Communications. In: Fifth Workshop on the Economics of Information Security (2006)
11. Crosby, J.: Challenges and Opportunities in Identity Assurance, HM Treasury (2008), http://www.hm-treasury.gov.uk/media/6/7/identity_assurance060308.pdf
12. Schlicht, E.: Isolation and Aggregation in Economics. Springer, New York (1985)
13. Information Technology Association of America: Identity Management: Building Trust, Mitigating Risks, Balancing Rights. White Paper (2005), http://www.itaa.org/news/docs/idmgmtwhitepaper.pdf
14. BBC: Brown apologises for records loss (2007), http://news.bbc.co.uk/1/hi/uk_politics/7104945.stm
15. Collins, T.: HMRC's Missing Child Benefit CDs - What Went Wrong and Lessons for NPfIT and ID cards. Computer Weekly (2007), http://www.computerweekly.com/blogs/tony_collins/2007/11/hmrcs-missing-child-benefit-cd-1.html
16. Wang, R.Y., Allen, T.J., Harris, W., Madnick, S.E.: An Information Product Approach for Total Information Awareness. MIT Sloan Working Paper No. 4407-02; CISL No. 2002-15 (November 2002), http://ssrn.com/abstract=376820
17. Straub, D.W.: Effective IS Security: An Empirical Study. Information Systems Research 1(3), 255–276 (1990)
18. Lucas, H., Olson, M.: The Impact of Information Technology On Organizational Flexibility. IOMS: Information Systems Working Papers, New York University, IS-93-49 (1993)
19. Wang, L., Wijesekera, D., Jajodia, S.: Cardinality-based inference control in data cubes. Journal of Computer Security 12(5), 655–692 (2004)
20. Sweeney, L.: k-anonymity: A Model for Protecting Privacy. International Journal on Uncertainty, Fuzziness and Knowledge-Based Systems 10(5), 557–570 (2002)
21. Bishop, M.: Computer Security: Art and Science. Addison-Wesley, New York (2002)
22. Katos, V., Patel, A.: A Partial Equilibrium View on Security and Privacy. Information Management & Computer Security 16(1), 74–83 (2008)

23. Inness, J.: Privacy, Intimacy and Isolation. Oxford University Press, Oxford (1992)
24. Branson, W.H., Litvack, J.M.: Macroeconomics, 2nd edn. Harper & Row, New York (1981)
25. Dornbush, R., Fischer, S.: Macroeconomics, 7th edn. McGraw-Hill, New York (1998)
26. Grossklags, J., Acquisti, A.: When 25 cents is too much: An experiment on willingness-to-sell and willingness-to-protect personal information. In: Proc. of the 6th Workshop on Economics and Information Security, Pittsburgh, USA (2007)
27. Schoderbek, P.P., Schoderbek, C.G., Kefalas, A.G.: Management Systems: Conceptual Considerations, Irwin, Boston (1990)

A Notation for Policies Using Feature Structures

Kunihiko Fujita and Yasuyuki Tsukada

NTT Communication Science Laboratories, NTT Corporation
3-1, Morinosato Wakamiya, Atsugi-shi, Kanagawa, 243-0198 Japan
{fujita,tsukada}@theory.brl.ntt.co.jp

Abstract. New security and privacy enhancing technologies are demanded in the new information and communication environments where a huge number of computers interact with each other in a distributed and ad hoc manner to access various resources. In this paper, we focus on access control because this is the underlying core technology to enforce security and privacy. Access control decides permit or deny according to access control policies. Since notations of policies are specialized in each system, it is difficult to ensure consistency of policies that are stated in different notations. In this paper, we propose a readable notation for policies by adopting the concept of feature structures, which has mainly been used for parsing in natural language processing. Our proposed notation is also logically well-founded, which guarantees strict access control decisions, and expressive in that it returns not only a binary value of permit or deny but also various result values through the application of partial order relations of the security risk level. We illustrate the effectiveness of our proposed method using examples from P3P.

1 Introduction

Access control is a mechanism that decides whether access to objects, such as information and contents, should be permitted or denied. The permit or deny criteria are set as access control policies. If these policies contain mistakes, a query that should be denied may be permitted. The complexity of policies is growing with today's large-scale information systems. Policies must keep up with environmental changes, such as changes in management and the transfer of members. As policies become larger and more complex, modifying them can result in unanticipated changes [1]. Moreover, since the notations of policies are specialized in each system, policy administrators must be proficient in all notations to maintain the policies. It is difficult to ensure consistency of policies that are stated in different notations.

In this paper, we propose an access control model based on *feature structures*. Feature structure notation has mainly been used for parsing in natural language processing, where it makes it possible to express a huge number of grammar rules clearly. Essentially, feature structures are sets of attributes. Each attribute is composed of a label/value pair. A value may be an atomic symbol but may also be a nested feature structure. We introduce a readable notation for policies by

J. Garcia-Alfaro et al. (Eds.): DPM 2010 and SETOP 2010, LNCS 6514, pp. 140–154, 2011.

adopting feature structures to facilitate policy maintenance. Since our proposed notation will improve the readability of policies, ease of maintenance will be enhanced.

Furthermore, our proposed notation is logically well-founded. To decide permit or deny strictly, access control models based on mathematical logic have been developed in the literature (See [4] for example). One of the advantages of mathematical-logic-based access control is that decisions can be proved to be strict. In our proposed method, access control decisions are accomplished with an operation called *unification* of feature structures. The unification procedure proposed in this paper first converts feature structures into logical formulas in accordance with rules and then reduces the logical formulas in accordance with the laws of equivalence to guarantee strict access control decisions.

Moreover, our proposed notation is expressive. Fundamentally, a result of an access control decision is a binary value of permit or deny. Since this "take it or leave it" approach is too limited, there is an increasing demand for various result values such as "permit for a specific part of a query" and "permit with an obligation for executing a required process". In our proposed method, results of access control decisions can not only be permit or deny but also can be permit for a specific part of queries. This is achieved by applying the lattice-based access control (LBAC) models [3] [10].

To illustrate the effectiveness of the proposed method, we apply it to the Platform for Privacy Preferences (P3P) [12] [2], an XML language designed to specify how websites intend to handle information they collect about their visitors.

The rest of this paper is organized as follows. Section 2 explains feature structures and unification. Section 3 introduces the proposed notation for policies using feature structures, the conversion rule from feature structures to the logical formulas, and the process of reasoning about logical formulas. In Section 4, we apply the proposed method to P3P as a case study. Section 5 discusses related work. Section 6 concludes the paper and summarizes future work.

2 Feature Structures

Feature structures are data structures used in unification-based grammar and have been used for parsing in natural language processing. With feature structures, it becomes possible to express a huge number of grammatical rules clearly and parse by means of an operation called unification. A feature structure is essentially a set of pairs comprising a feature label and its value. Feature labels show attributes, such as grammatical person, number, and tense. They are usually written in the following matrix form:

$$\begin{bmatrix} person: & \text{third} \\ number: & \text{plural} \end{bmatrix}$$

This expresses the pronoun of the third-person plural, "they", for example. The left column of the matrix lists labels and the right column lists correspondent values. The first line of the matrix shows that the value of the label *person* is third and the second line shows that the value of the label *number* is plural.

Moreover, a value is acceptable in not only an atomic symbol but also in a set of atomic symbols as in the following example:

$$\begin{bmatrix} person: & \{\text{first}, \text{second}, \text{third}\} \\ number: & \{\text{singular}, \text{plural}\} \\ tense: & \text{past} \end{bmatrix}$$

This expresses a verb in the past tense, "walked", for example. The first line of the matrix shows that the value of the label *person* can be first, second, or third. The second line shows that the value of the label *number* can be singular or plural. The third line shows that the value of the label *tense* is past. A value can be a feature structure as well.

A verb of third-person singular present tense, "walks", for example, expressed in feature structures is

$$\begin{bmatrix} person: & \text{third} \\ number: & \text{singular} \\ tense: & \text{present} \end{bmatrix}$$

The operation for composing one big component from more than one component is the unification of feature structures. Let the unification operator be $\otimes$, then we have

$$\begin{bmatrix} person: & \text{third} \\ number: & \text{plural} \end{bmatrix} \otimes \begin{bmatrix} person: & \{\text{first}, \text{second}, \text{third}\} \\ number: & \{\text{singular}, \text{plural}\} \\ tense: & \text{past} \end{bmatrix} = \begin{bmatrix} person: & \text{third} \\ number: & \text{plural} \\ tense: & \text{past} \end{bmatrix}$$

This unification example expresses that the sentence "They walked" is grammatically correct since the unification succeeds. The unification result shows the attribute of the sentence.

On the other hand, $FAIL$ shows failure of the unification such that

$$\begin{bmatrix} person: & \text{third} \\ number: & \text{plural} \end{bmatrix} \otimes \begin{bmatrix} person: & \text{third} \\ number: & \text{singular} \\ tense: & \text{present} \end{bmatrix} = FAIL$$

In this case, the unification fails because the label *number* is assigned the different values plural and singular. This unification example expresses that a sentence "They walks" is grammatically incorrect since the unification fails.

3 Applying Feature Structures to Policies

In this section, first we explain the process for access control. Then, we illustrate the notation for policies using feature structures. Lastly, since formally defining the unification is not straightforward because of the form of the feature structures, we define the logical system FDLP (Feature Description Logic for Policy).

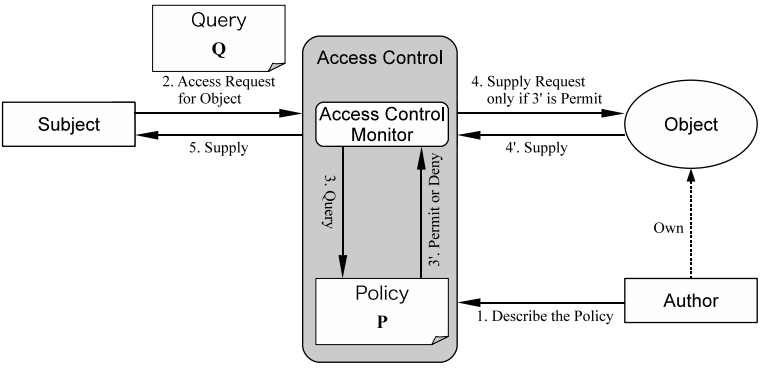


Fig. 1. A simple access control mechanism

3.1 Access Control

Figure 1 shows a simple access control mechanism. The process is as follows:

1. The agent who owns the rights to the object (Author) writes a policy **P** as a condition for permitting the use of the object.
2. The agent who wants to use the object (Subject) sends to an access control a query **Q** as an access request for the use of the object.
3. The control section decides whether **Q** satisfies **P**.
4. The control section sends to the object a supply request only if **P** returns permit.
5. The control section provides the object to the subject.

3.2 Policy Expression by Feature Structures

Policies contain definitions of what rights an author permits a subject to execute on a certain object. Policies might contain conditions that must be satisfied before access is permitted. Accordingly, the following elements should be included in policies:

Author, Subject, Object, Right, and Condition.

Moreover, if a subject's query satisfies conditions written in a policy, the use of an object is permitted. In other words, queries must contain elements that correspond to elements of policies to make access control decisions of permit or deny. Let a policy be expressed in feature structures by $\mathbf{P}^{\mathrm{FS}}$ and a query by $\mathbf{Q}^{\mathrm{FS}}$. Both $\mathbf{P}^{\mathrm{FS}}$ and $\mathbf{Q}^{\mathrm{FS}}$ have the following form:

$$\begin{bmatrix} auth: & \mathrm{v}^{auth} \\ subj: & \mathrm{v}^{subj} \\ obj: & \mathrm{v}^{obj} \\ right: & \mathrm{v}^{right} \\ cond: & \mathrm{v}^{cond} \end{bmatrix}$$

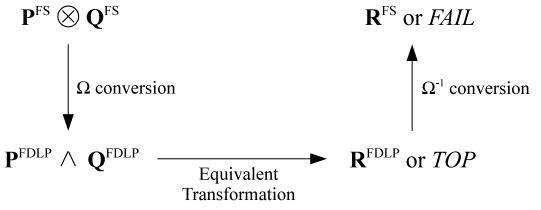


Fig. 2. The concept of conversions and transformation

where *auth* is a feature label of author, *subj* denotes subject, *obj* denotes object, *right* denotes a right, and *cond* denotes a condition. The v^{auth}, v^{subj}, v^{obj}, v^{right}, and v^{cond} are the values of the corresponding feature labels. The set of values of each feature label is either an ordinary set or a partially ordered set. The order structure depends on each specific application domain. This will be explained in detail in section 3.4.

3.3 Converting Feature Structures to Logical Formulas

In our method, policies and queries are expressed by feature structures and the decisions of permit or deny correspond to whether the unifications of feature structures succeed or not.

Feature structures are easy-to-understand forms, while logical formulas have merits such as providing a way to perform unification precisely [6]. Note that no information is added or subtracted by rewriting the feature structure as a logical formula. These two forms may be regarded as notational variants for expressing the same facts. While feature structures seem to be a more appealing and more natural notation for displaying linguistic descriptions, logical formulas provide a precise interpretation that can be useful for computational and mathematical purposes.

For the above-mentioned reasons, the interfaces between human and access-control system are described in feature structures and the operations, such as unification, are described in logical formulas that are converted from feature structures (Figure 2).

In the following, we define the syntax of FDLP logical formulas of policies and queries that are converted from policies expressed in feature structures, and define rule Ω for conversion from feature structures to logical formulas and rule Ω^{-1} for the reverse.

Syntax of FDLP. We formally define the domain FDLP of logical formulas, which describe feature structures according to the following syntax of well-formed formulas.

NIL	denoting no information
TOP	denoting inconsistent information
a	where $a \in A$, to describe atomic values

$$
\begin{array}{rl}
l : \phi & \text{where } l \in L \text{ and } \phi \in \mathsf{FDLP} \\
\phi \wedge \psi & \text{denoting the conjunction of } \phi, \psi \in \mathsf{FDLP} \\
\phi \vee \psi & \text{denoting the disjunction of } \phi, \psi \in \mathsf{FDLP},
\end{array}
$$

where the formulas NIL and TOP are intended to convey "no information" and "inconsistent information", respectively. Thus, NIL corresponds to a unification variable and TOP corresponds to a unification failure. L and A are the finite sets of symbols for feature labels and values, respectively. The formula $l : \phi$ indicates that a value has attribute l, which is itself a value satisfying the condition ϕ. Conjunction and disjunction will have their ordinary logical meaning as operators in formulas. An interesting result is that conjunction can be used to describe unification. Unifying two structures requires finding a structure that has all features of both structures; the conjunction of two formulas describes the structures that satisfy all conditions of both formulas.

Conversion Rules. Rules Ω and Ω^{-1} for converting feature structures to logical formulas and vice versa are shown in Figure 3 and 4, respectively. Precisely, Ω^{-1} is defined for certain "normal forms" of FDLP logical formulas. This is discussed in the next section.

3.4 Access Control Decision and Reasoning Range of Use

In this section, we define the laws of equivalence for FDLP and show that the access control decision and reasoning range of use can be executed on our framework.

First, we introduce the laws of equivalence for FDLP, which are listed in Figure 5. Here, $\phi, \psi, \chi \in \mathsf{FDLP}$ and $a, b, c \in ACl \in L$ are assumed. The "inf X" denotes the greatest lower bound of set X. Here, we consider the lower bound with respect to the specific partially ordered set that includes X as a subset.

Each FDLP formula can be transformed by applying these laws. In particular, in the applications of equations of Failure, Conjunction, and Disjunction, the transformations from the left side to the right side are only allowed. Since

$$
\Omega\left(\begin{bmatrix} l_1 : \mathrm{v}_1 \\ \vdots \quad \vdots \\ l_n : \mathrm{v}_n \end{bmatrix}\right) = \Omega(l_1 : v_1) \wedge \cdots \wedge \Omega(l_n : v_n)
$$

$$
\Omega(l : v) = \begin{cases}
l : (a_1 \vee \cdots \vee a_m) & v = \{a_1, \cdots a_m\} \\
 & (v \text{ is a set of atomic value.}) \\
l : \Omega(v) & v = \begin{bmatrix} l_j : \mathrm{v}_j \\ \vdots \quad \vdots \\ l_k : \mathrm{v}_k \end{bmatrix} \\
 & (v \text{ is a feature structure}) \\
l : NIL & v = NIL \\
 & (v \text{ has no information}) \\
l : v & \text{otherwise} \\
 & (v \text{ is an atomic value})
\end{cases}
$$

Fig. 3. Rule Ω for converting feature structures to FDLP

$$\Omega^{-1}(TOP) = FAIL$$

$$\Omega^{-1}(l_1 : v_1 \wedge \cdots \wedge l_n : v_n) = \begin{bmatrix} \Omega^{-1}(l_1 : \mathrm{v}_1) \\ \vdots \quad \vdots \\ \Omega^{-1}(l_n : \mathrm{v}_n) \end{bmatrix}$$

$$\Omega^{-1}(l : v) = \begin{cases} l : \{a_1, \cdots, a_m\} & v = a_1 \vee \cdots \vee a_m \\ & (v \text{ is a disjunction of atomic values}) \\ l : \Omega^{-1}(v) & v = l_j : v_j \wedge \cdots \wedge l_k : v_k \\ & (v \text{ is a conjunction of pairs} \\ & \quad \text{of a label and a value}) \\ l : NIL & v = NIL \\ & (v \text{ has no information}) \\ l : v & \text{otherwise} \\ & (v \text{ is an atomic value}) \end{cases}$$

Fig. 4. Rule Ω^{-1} for converting FDLP to feature structures

Failure

$$l : TOP = TOP \quad (1)$$

Conjunction

$$\phi \wedge TOP = TOP \quad (2)$$

$$\phi \wedge NIL = \phi \quad (3)$$

$$a \wedge b = \begin{cases} \inf\{a, b\} & \text{if } \inf\{a, b\} \text{ exists} \\ TOP & \text{otherwise} \end{cases} \quad (4)$$

$$a \wedge l : \phi = TOP \quad (5)$$

$$l : \phi \wedge l : \psi = l : (\phi \wedge \psi) \quad (6)$$

Disjunction

$$\phi \vee TOP = \phi \quad (7)$$

$$\phi \vee NIL = NIL \quad (8)$$

$$l : \phi \vee l : \psi = l : (\phi \vee \psi) \quad (9)$$

Commutative

$$\phi \wedge \psi = \psi \wedge \phi \quad (10)$$

$$\phi \vee \psi = \psi \vee \phi \quad (11)$$

Associative

$$(\phi \wedge \psi) \wedge \chi = \phi \wedge (\psi \wedge \chi) \quad (12)$$

$$(\phi \vee \psi) \vee \chi = \phi \vee (\psi \vee \chi) \quad (13)$$

Idempotent

$$\phi \wedge \phi = \phi \quad (14)$$

$$\phi \vee \phi = \phi \quad (15)$$

Distributive

$$(\phi \vee \psi) \wedge \chi = (\phi \wedge \chi) \vee (\psi \wedge \chi) \quad (16)$$

$$(\phi \wedge \psi) \vee \chi = (\phi \vee \chi) \wedge (\psi \vee \chi) \quad (17)$$

Absorption

$$(\phi \wedge \psi) \vee \phi = \phi \quad (18)$$

$$(\phi \vee \psi) \wedge \phi = \phi \quad (19)$$

Fig. 5. Laws of equivalence for FDLP

the number of feature structures and atomic values is reduced by applying the equations of Failure, Conjunction, and Disjunction, we can see that every FDLP formula can be transformed to a specific form that belongs to the domain of Ω^{-1}.

The laws of equivalence for FDLP extend those for FDL (Feature Description Logic) by Kasper and Rounds [6] to handle the security risk level relation. The security risk level is a security-risk criterion with which certain elements of a domain of policy components, such as data, user, and purpose, are provided relatively.

Definition 1 (Security Risk Level Relation). *Let s_i and s_j be elements of a domain of policy components D. We write $s_i \leq_D s_j$ if the relation that a notion s_j indicates a higher security risk level than that of s_i holds.* □

For example, we examine four elements – "indefinitely", "no retention","legal requirement", and "business practices" – with respect to the term for the grant of permission for use, and these are the elements of domain T. From the author's viewpoint, the longer the term of use is, the higher the security risk level becomes intuitively. We define security risk level ordering $\leq_T$ on these elements, with "indefinitely" having the greatest value, "no retention" the least value, and "legal requirement" and "business practices" mutually incomparable values but less than "indefinitely" and greater than "no retention". Therefore, the relations

$$\begin{aligned} \text{legal requirement} &\leq_T \text{indefinitely} \\ \text{business practices} &\leq_T \text{indefinitely} \\ \text{no retention} &\leq_T \text{legal requirement} \\ \text{no retention} &\leq_T \text{business practices} \\ \text{no retention} &\leq_T \text{indefinitely} \end{aligned}$$

hold.

The concept of the enhancement we propose in this paper is based on LBAC models [3] [10]. In LBAC, the security risk level may also be expressed in terms of a lattice. In our model, a subject is permitted to access an object at at most the security risk level of the object.

Definition 2 (Accessible Security Risk Level). *Let s_i and s_j be the security risk levels of a subject and an object, respectively, and be elements of a domain of policy components $(D, \leq_D)$. The subject can access the object at the security risk level* $\inf\{s_i, s_j\}$ *(if it exists).* □

For example, the above-mentioned $(T, \leq_T)$ is a lattice because for any two elements a and b of T, the set $\{a, b\}$ has a supremum and an infimum. If the security risk levels "legal requirement" and "indefinitely" are assigned to a subject and an object, respectively, the subject can access the object at "legal requirement". Similarly, if the security risk levels "legal requirement" and "business practices" are assigned to a subject and an object, respectively, the subject can access the object at "no retention". We condense the above-mentioned notions into the formula (4) in the laws of equivalence (Figure 5).

The access control decision and the reasoning of the range of use are explained as follows. Let $\mathbf{P}^{\mathrm{FDLP}}$ and $\mathbf{Q}^{\mathrm{FDLP}}$ be a policy and a query expressed in FDLP, respectively. If $\mathbf{P}^{\mathrm{FDLP}} \wedge \mathbf{Q}^{\mathrm{FDLP}} = TOP$ is reasoned as a result of the equivalent transformation, it expresses that a contradiction has occurred in the unification process of $\mathbf{P}^{\mathrm{FDLP}}$ and $\mathbf{Q}^{\mathrm{FDLP}}$, i.e., the unification fails. This means the query $\mathbf{Q}$ does not satisfy the conditions written in the policy $\mathbf{P}$ and permission for use is not granted. If $\mathbf{P}^{\mathrm{FDLP}} \wedge \mathbf{Q}^{\mathrm{FDLP}} = \mathbf{R}^{\mathrm{FDLP}}$ is reasoned, it expresses that the unification succeeds and the result is $\mathbf{R}^{\mathrm{FDLP}}$. This means the query $\mathbf{Q}$ satisfies the conditions written in policy $\mathbf{P}$ and permission for use is granted under the condition written in $\mathbf{R}^{\mathrm{FDLP}}$.

4 Applying the Proposed Method to P3P

In this section, we show how the proposed method can be applied to P3P to prove the usefulness of applying feature structures to policies. First, we explain P3P. Then, we show examples to prove that P3P policies can be expressed in feature structures and our method can decide permit or deny and reason the range of use.

4.1 P3P

P3P enables websites to express their privacy policies in a standard XML format that can be retrieved automatically and interpreted easily by user agents. Usually, a website publishes its privacy policy in a well-known location so that client software acting on the user's behalf can examine the policy and compare it to the preferences the user has configured to express how user's data is to be used. If the policy meets the client's preferences, the software agent approves the transaction, and the user continues browsing without any noticeable interruption.

The most important elements in P3P policies are the "Statement" elements, which include the following child elements.

- Data-Group: Contains one or more DATA elements. DATA elements are used to describe the type of data that a website collects (e.g., mail addresses and telephone numbers).
- Purpose: Contains one or more purposes of data collection or uses of data. Websites must classify the purposes for which they use data into one or more of the purposes specified below. The Purpose element must contain one or more of the following sub-elements.
  - `CUR` Completion and support of activity for which data was provided.
  - `ADM` Website and system administration.
  - `DEV` Research and development.
  - `TAI` One-time tailoring.
  - `PSA` Pseudonymous analysis.
  - `PSD` Pseudonymous decision.
  - `IVA` Individual analysis.
  - `IVD` Individual decision.
  - `CON` Contacting visitors for marketing of services or products.
  - `HIS` Historical preservation.
  - `TEL` Contacting visitors for marketing of services or products via telephone.
  - `OPT` Other uses.
- Recipient: Contains one or more recipients of the collected data. Websites must classify their recipients into one or more of the six recipients specified below. The Recipient element must contain one or more of the following sub-elements.
  - `OUR` Ourselves and/or entities acting as our agents or entities for whom we are acting as an agent.
  - `DEL` Delivery services possibly following different practices.
  - `SAM` Legal entities following our practices.
  - `OTR` Legal entities following different practices.

Table 1. Example P3P policy snippets

website A

```
<STATEMENT>
  <DATA-GROUP>
    <DATA REF="#USER.MAIL-ADDRESS.GIVEN"/>
  </DATA-GROUP>
  <PURPOSE>
     <TAILORING/> <CONTACT/>
  </PURPOSE>
  <RECIPIENT>
     <UNRELATED/> <SAME/>
  </RECIPIENT>
  <RETENTION>
     <BUSINESS-PRACTICES/>
  </RETENTION>
</STATEMENT>
```

website B

```
<STATEMENT>
  <DATA-GROUP>
    <DATA REF="#USER.MAIL-ADDRESS.GIVEN"/>
  </DATA-GROUP>
  <PURPOSE>
     <TAILORING/> <PSEUDO-ANALYSIS/>
  </PURPOSE>
  <RECIPIENT>
     <DELIVERY/>
  </RECIPIENT>
  <RETENTION>
     <NO-RETENTION/>
  </RETENTION>
</STATEMENT>
```

`UNR` Unrelated third parties.
`PUB` Public fora.

– Retention: Indicates the kind of retention policy that applies to the data referenced in that statement. The Retention element must contain one sub-element of the following.
`NOR` Information is not retained for more than the brief period of time necessary to make use of it during the course of a single online interaction.
`STP` Information is retained to meet the stated purpose.
`LEG` Information is retained to meet a stated purpose, but the retention period is longer because of a legal requirement or liability.
`BUS` Information is retained under a service provider's stated business practices.
`IND` Information is retained for an indeterminate period of time.

Examples of P3P policies are shown in Table 1.

Here, we consider how "Purpose", "Recipient" and "Retention" can be regarded as partially ordered sets over the security risk level relation[1]. We denote the set of the sub-elements of "Purpose" by P, that of "Recipient" by R, and that of "Retention" by T (i.e., P, R, and T are the domains of policy components). P is the element for the purposes of data collection or uses of data, and it is difficult to order by the security risk level relation for such a qualitative element. In addition, the meanings of the sub-elements of P are fuzzy and mix notions of business relationships and policy [5]. On the other hand, R is the element for the recipients of the collected data, and the sub-elements of R can be classified as follows.

	recipient	our policy
`OUR`	A	○
`DEL`	B	×
`SAM`	B	○
`OTR`	B	×
`UNR`	B	×
`PUB`	C	×

[1] Although the other domains of policy components, such as data-group, may be a partially ordered set over the security risk level relation, we concentrate on the domains of "Purpose", "Recipient" and "Retention" because of space limitations.

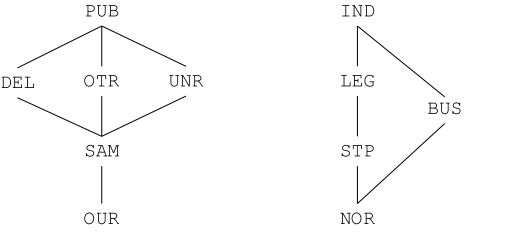


Fig. 6. Hasse Diagrams

The first column of the table classifies sub-elements of R into (A) our organization, (B) specific organizations that are outside of ours, and (C) no limitation. The second column indicates whether the recipient obeys our policy ($\circ$) or not ($\times$). For example, the security risk level of `SAM` is less than that of `DEL`, `OTR` and `UNR`, since `SAM` obeys our policy and the others do not. We arrange the table and show the Hasse diagram of R in Figure 6 (1). T is the element for the retention of collected data, and as shown in section 3.4, the security risk level relations with `IND`, `NOR`, `LEG`, and `BUS` hold. We arrange the relations of these four with `STP` and show the Hasse diagram of T in Figure 6 (2).

4.2 Expressing P3P Policies in Feature Structures

We express P3P policies in feature structures and define in a formal way the method for deciding whether permission for use should be granted. In applying our method to P3P, a website visitor corresponds to the author, the visitor's private information corresponds to the object, and a website corresponds to a subject because websites try to obtain permission to use the visitor's private information. A right corresponds to the use of the visitor's private information, and conditions correspond to the purposes (P), recipients (R), and retention (T) [7]. The author's policy ($\mathbf{P}$) and subject's query ($\mathbf{Q}$) can be expressed in feature structures on the framework described in section 3.2 as follows.

$$\mathbf{P}^{\mathrm{FS}} = \begin{bmatrix} auth: \text{ visitor} \\ subj: \ NIL \\ obj: \begin{bmatrix} d_1: \text{ private\_info}_1 \\ \vdots \qquad\qquad \vdots \\ d_n: \text{ private\_info}_n \end{bmatrix} \\ right: \text{use} \\ cond: \begin{bmatrix} P: \mathrm{v}^P \\ R: \mathrm{v}^R \\ T: \mathrm{v}^T \end{bmatrix} \end{bmatrix} \qquad \mathbf{Q}^{\mathrm{FS}} = \begin{bmatrix} auth: \ NIL \\ subj: \text{ website} \\ obj: \begin{bmatrix} d_1: \text{ private\_info}_1 \\ \vdots \qquad\qquad \vdots \\ d_n: \text{ private\_info}_n \end{bmatrix} \\ right: \text{use} \\ cond: \begin{bmatrix} P: \mathrm{v}^P \\ R: \mathrm{v}^R \\ T: \mathrm{v}^T \end{bmatrix} \end{bmatrix}$$

Since a website visitor's policy is applied to all websites, the value of *subj* is NIL in $\mathbf{P}^{\mathrm{FS}}$. Similarly, since a website's query is applied to all of visitors, the value *auth* is NIL in $\mathbf{Q}^{\mathrm{FS}}$. We assume such settings after due consideration of the usual use of P3P.

The queries of websites A and B shown in Table 1 expressed in feature structures are as follows.

$$\mathbf{Q}^{\mathrm{FS}}_{\mathrm{website_A}} = \begin{bmatrix} auth\colon NIL \\ subj\colon \mathrm{website_A} \\ obj\colon \ [\, d_1\colon NIL\,] \\ right\colon \mathrm{use} \\ cond\colon \begin{bmatrix} P\colon \{\mathtt{TAI}, \mathtt{CON}\} \\ R\colon \{\mathtt{UNR}, \mathtt{SAM}\} \\ T\colon \mathtt{BUS} \end{bmatrix} \end{bmatrix} \qquad \mathbf{Q}^{\mathrm{FS}}_{\mathrm{website_B}} = \begin{bmatrix} auth\colon NIL \\ subj\colon \mathrm{website_B} \\ obj\colon \ [\, d_1\colon NIL\,] \\ right\colon \mathrm{use} \\ cond\colon \begin{bmatrix} P\colon \{\mathtt{TAI}, \mathtt{PSA}\} \\ R\colon \mathtt{DEL} \\ T\colon \mathtt{NOR} \end{bmatrix} \end{bmatrix}$$

Here, we assume Alice is a visitor and her policy expressed in a feature structure is as follows.

$$\mathbf{P}^{\mathrm{FS}}_{\mathrm{Alice}} = \begin{bmatrix} auth\colon \mathrm{Alice} \\ subj\colon NIL \\ obj\colon \ [\, d_1\colon \mathrm{alice@foo.bar.jp}\,] \\ right\colon \mathrm{use} \\ cond\colon \begin{bmatrix} P\colon \{\mathtt{CON}, \mathtt{TEL}\} \\ R\colon \{\mathtt{OTR}, \mathtt{UNR}, \mathtt{SAM}\} \\ T\colon \mathtt{LEG} \end{bmatrix} \end{bmatrix}$$

These queries and policy are converted to the following FDLP formulas by virtue of the conversion rules Ω.

$$\begin{aligned} \mathbf{Q}^{\mathrm{FDLP}}_{\mathrm{website_A}} = {} & auth : NIL \\ & \wedge\ subj\colon \mathrm{website_A} \\ & \wedge\ obj\colon (d_1 : NIL) \\ & \wedge\ right\colon \mathrm{use} \\ & \wedge\ cond\colon (P\colon (\mathtt{TAI} \vee \mathtt{CON}) \\ & \qquad \wedge\ R\colon (\mathtt{UNR} \vee \mathtt{SAM}) \\ & \qquad \wedge\ T\colon \mathtt{BUS}) \end{aligned} \qquad \begin{aligned} \mathbf{Q}^{\mathrm{FDLP}}_{\mathrm{website_B}} = {} & auth\colon NIL \\ & \wedge\ subj\colon \mathrm{website_B} \\ & \wedge\ obj\colon (d_1 : NIL) \\ & \wedge\ right\colon \mathrm{use} \\ & \wedge\ cond\colon (P\colon (\mathtt{TAI} \vee \mathtt{PSA}) \\ & \qquad \wedge\ R\colon \mathtt{DEL} \\ & \qquad \wedge\ T\colon \mathtt{NOR}) \end{aligned}$$

$$\begin{aligned} \mathbf{P}^{\mathrm{FDLP}}_{\mathrm{Alice}} = {} & auth\colon \mathrm{Alice} \\ & \wedge\ subj\colon NIL \\ & \wedge\ obj\colon (d_1\colon \mathrm{alice@foo.bar.jp}) \\ & \wedge\ right\colon \mathrm{use} \\ & \wedge\ cond\colon (P\colon (\mathtt{CON} \vee \mathtt{TEL}) \\ & \qquad \wedge\ R\colon (\mathtt{OTR} \vee \mathtt{UNR} \vee \mathtt{SAM}) \\ & \qquad \wedge\ T\colon \mathtt{LEG}) \end{aligned}$$

Utilizing equivalent transformation defined in Figure 5, we reason the decisions of permission for use of the private information of Alice by the websites A and B.

$$\mathbf{P}^{\mathrm{FDLP}}_{\mathrm{Alice}} \wedge \mathbf{Q}^{\mathrm{FDLP}}_{\mathrm{website_A}} = \begin{array}{l} auth\colon \mathrm{Alice} \\ \wedge\ subj\colon \mathrm{website_A} \\ \wedge\ obj\colon (d_1\colon \mathrm{alice@foo.bar.jp}) \\ \wedge\ right\colon \mathrm{use} \\ \wedge\ cond\colon (P\colon \mathtt{CON} \\ \qquad \wedge\ R\colon (\mathtt{UNR} \vee \mathtt{SAM}) \\ \qquad \wedge\ T\colon \mathtt{NOR}) \end{array}$$

$$\mathbf{P}^{\mathrm{FDLP}}_{\mathrm{Alice}} \wedge \mathbf{Q}^{\mathrm{FDLP}}_{\mathrm{website_B}} = TOP$$

Lastly, we convert these results into feature structures by the conversion rules Ω^{-1}.

$$\mathbf{P}^{\mathrm{FS}}_{\mathrm{Alice}} \otimes \mathbf{Q}^{\mathrm{FS}}_{\mathrm{website_A}} = \left[\begin{array}{ll} auth\colon & \mathrm{Alice} \\ subj\colon & \mathrm{website_A} \\ obj\colon & \left[d_1\colon \mathrm{alice@foo.bar.jp}\right] \\ right\colon & \mathrm{use} \\ cond\colon & \left[\begin{array}{l} P\colon \mathtt{CON} \\ R\colon \{\mathtt{UNR}, \mathtt{SAM}\} \\ T\colon \mathtt{NOR} \end{array}\right] \end{array}\right]$$

$$\mathbf{P}^{\mathrm{FS}}_{\mathrm{Alice}} \otimes \mathbf{Q}^{\mathrm{FS}}_{\mathrm{website_B}} = FAIL$$

5 Related Work

May et al. offered flexible policy comparison metrics by adopting concepts from the process calculus literature [7]. They call the policy relations they devised *strong* and *weak* licensing. They use a running example in P3P. Their research motivated us to seek a flexible policy comparison metrics by introducing the security risk level relation. Although improving policy readability was out of their scope, it is the main topic of our research.

Privacy policy languages, such as P3P, currently designed to protect users' privacy follow a "take it or leave it" approach that is inflexible. Walker et al. proposed a privacy policy negotiation protocol that is based on an "Or Best Offer" (OBO) negotiation style [11]. In other words, they supported various results of access control decisions, i.e., not only permit or deny but also permit for a specific part of a query, which we also achieved in this work by applying partial order relations of the security risk level. In their method, a client and a server have to define their strategy to accomplish more sophisticated policy negotiation. Our method aims to lighten the workload for policy maintenance; therefore, we adopt a simple approach.

Ni et al. introduced an access control model named P-RBAC (Privacy-aware RBAC) that extends the RBAC (Role-based access control) model in order to provide full support for expressing highly complex privacy-related policies, taking into account features like purposes and obligations [8]. They introduced the notions of role hierarchy, data hierarchy, and purpose hierarchy and their notions are closely related to our notion of security risk level relation. However, the main

topic of their study is the detection of conflicts in permission assignments that are rules in policies. This differs from improving policy readability, which is our main topic.

6 Conclusion and Future Work

To introduce readable notation for policies, we propose an access control model using feature structures. In the proposed model, strict access control decisions are made by converting feature structures into logical formulas and reasoning them on the defined laws of equivalence. Moreover, access control decisions can be included in not only the binary value of permit or deny but also in the permit for a specific part of a query. To illustrate the effectiveness of the proposed method, we applied it to P3P.

When the proposed method is applied to actual access control techniques, the method must be customized to conform to each technique, as shown by the application example to P3P. As examples, we examine DAC (Discretionary Access Control), MAC (Mandatory Access Control), and RBAC briefly. DAC is a simple technique with which an author can discretionary give subjects rights of access to objects, and the proposed method can be applied almost as it is. In the case of MAC, an author can give rights of access only within a range that a system has decided. Even if an author permits access, access will be denied when a system denies it. When the proposed method is applied to MAC, an algorithm at the meta level must be developed to combine policies. In the case of RBAC, because a role is dealt with as a subject of a special kind, a means to check whether roles should be assigned to subjects is needed.

XACML allows a policy to contain multiple sub-policies, and the result of the policy on a query is determined by combining the results of the sub-policies according to some algorithms. As future work, we plan to develop such a policy-combining algorithm for the proposed model to describe more complex privacy policies such as Facebook's.

In this paper, we introduce feature structures as a readable notation. Since there have been no evaluations of readability, we plan to conduct such an evaluation. The evaluation method under consideration is to present policies and queries of various notations, including feature structures and FDLP, to test subjects, and analyze the subjects' suppositions of the results of the queries. A readable notation should support inferring correct suppositions.

To realize strict access control decisions, our proposed method converts policies expressed with feature structures to those expressed with FDLP, which follows the method of Kasper and Rounds [6]. However, it may be possible to realize strict access control decisions; either directly on feature structures or by using a more conventional logic, or even relational algebra. Comparing these approaches with our method is also future work.

Other future work includes proving the confluence and termination of equational transformation described in Figure 5. Our work paves the way for considering more general policy representation languages like XACML [9], which is becoming increasingly important in the security domain.

Acknowledgment

We would like to thank the reviewers for their comments that help improve the manuscript. We also thank Satoshi Tojo of Japan Advanced Institute of Science and Technology, and Yoshifumi Manabe, Ken Mano and Hideki Sakurada of the Innovative Communication Laboratory of NTT Communication Science Laboratories for their accurate comments and advice.

References

1. Capretta, V., Stepien, B., Felty, A., Matwin, S.: Formal correctness of conflict detection for firewalls. In: Proceedings of the 2007 ACM Workshop on Formal Methods in Security Engineering, FMSE 2007, pp. 22–30. ACM, New York (2007)
2. Cranor, L.: P3P: Making privacy policies more useful. IEEE Security & Privacy 1(6), 50–55 (2003)
3. Denning, D.E.: A lattice model of secure information flow. ACM Commun. 19(5), 236–243 (1976)
4. Halpern, J.Y., Weissman, V.: Using first-order logic to reason about policies. ACM Trans. Inf. Syst. Secur. 11(4), 1–41 (2008)
5. Karjoth, G., Schunter, M., Herreweghen, E.V., Waidner, M.: Amending P3P for clearer privacy promises. In: Proceedings of the 14th International Workshop on Database and Expert Systems Applications, DEXA 2003, pp. 445–449. IEEE Computer Society, Washington, DC (2003)
6. Kasper, R.T., Rounds, W.C.: A logical semantics for feature structures. In: Proceedings of the 24th Annual Meeting on Association for Computational Linguistics, pp. 257–266. Association for Computational Linguistics, Morristown (1986)
7. May, M.J., Gunter, C.A., Lee, I., Zdancewic, S.: Strong and weak policy relations. In: Proceedings of the 2009 IEEE International Symposium on Policies for Distributed Systems and Networks, POLICY 2009, pp. 33–36. IEEE Computer Society, Washington, DC (2009)
8. Ni, Q., Trombetta, A., Bertino, E., Lobo, J.: Privacy-aware role based access control. In: Proceedings of the 12th ACM Symposium on Access Control Models and Technologies, SACMAT 2007, pp. 41–50. ACM, New York (2007)
9. Organization for the Advancement of Structured Information Standards (OASIS): Extensible Access Control Markup Language (XACML), http://xml.coverpages.org/xacml.html
10. Sandhu, R.S.: Lattice-based access control models. Computer 26(11), 9–19 (1993)
11. Walker, D.D., Mercer, E.G., Seamons, K.E.: Or best offer: A privacy policy negotiation protocol. In: Proceedings of the 2008 IEEE International Symposium on Policies for Distributed Systems and Networks, POLICY 2008, pp. 173–180. IEEE Computer Society, Washington, DC (2008)
12. World Wide Web Consortium (W3C): P3P: The Platform for Privacy Preferences, http://www.w3.org/P3P/

Securing P2P Storage with a Self-organizing Payment Scheme

Nouha Oualha[1] and Yves Roudier[2]

[1] Telecom ParisTech, Paris, France
Tel.: +33 (0)1 45 81 77 41; Fax: +33 (0)1 45 89 79 06
oualha@telecom-paristech.fr
[2] EURECOM, Sophia Antipolis, France
Tel.: +33 (0)4 93 00 81 18; Fax: +33 (0)4 93 00 82 00
roudier@eurecom.fr

Abstract. This paper describes how to establish trust for P2P storage using a payment-based scheme. This scheme relies on the monitoring of storage peers on a regular basis. The verification operations allow assessing peer behavior and eventually estimating their subsequent remuneration or punishment. The periodic verification operations are used to enforce the fair exchange of a payment against effective cooperative peer behavior. Payments are periodically provided to peers based on the verification results. Only cooperative peers are paid by data owners with the help of intermediates in the P2P network, thus accommodating peer churn. Moreover, our payment scheme does not require any centralized trusted authority to appropriately realize a large-scale system. Simulations in this paper evaluate the capability of the payment scheme to work as a sieve to filter out non cooperative peers.

Keywords: P2P storage, cooperation, Storage reliability, payment based scheme.

1 Introduction

The tremendous growth in the amount of data available in today networks and the trend towards increasing self-organization, as illustrated by large scale distributed systems and mobile ad hoc networks, generate a growing interest in peer-to-peer (P2P) data storage. Such an application exposes the data to new threats due to peers since any service deployed in self-organizing networks first and foremost relies on their willingness to cooperate. In this setting, selfishness not only covers explicit non cooperation, but also subtle forms of misbehavior whereby peers may discard some data they promised to store in order to optimize their resource usage or may try to escape their responsibility in contributing to the maintenance of the storage infrastructure.

Approaches to overcome non cooperating behaviors fall into two classes: reputation and payment (or remuneration) mechanisms. Reputation builds a long-term trust between peers based on a statistical history of their past interactions. On the other hand, payment-based mechanisms feature a more immediate

J. Garcia-Alfaro et al. (Eds.): DPM 2010 and SETOP 2010, LNCS 6514, pp. 155–169, 2011.

reward for a correct behavior making possible an explicit and discrete association of incentives and cooperation. With payments, peers cooperate because they are getting paid not because of a hypothetic reciprocation in the future as can be found with reputation systems.

Several payment schemes have been introduced to enable ad hoc packet forwarding ([1], [2]). It is however quite easy to determine individual and instantaneous rewards fostering cooperation in exchange of a packet forwarding operation, whereas data storage can only be deemed as correct at data retrieval time. Furthermore, it is useful in this case to be aware of a storage failure as soon as possible. Other payment schemes are not attached to a particular resource sharing application ([19], [20], [21], and [22]); but they unfortunately rather suppose the presence of a centralized authority to mediate peer payments, and they therefore may undermine the scalability of a P2P system.

The contributions of this paper are summarized in the following. A description of the problem statement (Section 2) that particularly defines our threat model and related work (Section 3). The design of a payment based scheme for mitigating the impact of selfishness towards P2P storage (Section 4). The proposed scheme relying on KARMA [14] for peer account administration is based on a cryptographic protocol that enforces the periodic fair-exchange of payment against cooperative behavior. An evaluation of that scheme (Section 5) based on simulation results demonstrating its capability to detect and punish non cooperative peers and an analysis of its security.

2 Problem Statement

P2P cooperative storage systems feature peers that store data for other peers; in counterpart, peers receive some disk space for the storage of their own data. Such systems face two major issues: reliability, since holder peers may crash or delete the data they store; and fairness, since peers may refuse to store data for others. The data stored can be periodically checked using one of the numerous verification protocols discussed in [4] and [5]. [4] discusses how verification can be mostly handled by verifier peers selected and appointed by the data owner to distribute the load of this task. The current paper discusses how to enable such a system using monetary compensations for the verification task.

Threat model. P2P storage relies on cooperation with unknown peers without prior trust relationships. However, peers may misbehave in various ways. In comparison with the study of selfish behaviors in routing over mobile ad-hoc network, we distinguish two types of selfishness:

- **Passive selfishness:** Passively selfish peers store their own data in the system but never concede disk space for the storage of other peers. This problem has long been referred to as free-riding. In a storage system like the one described in [4], free-riders will also refuse to perform data storage verifications although they will request other peers to check the correct storage of their own data.

– **Active selfishness:** Actively selfish peers accept to store data although they will not in effect store them for the agreed upon period of time but instead destroy them at some point to optimize their resource usage. Actively selfish peers also agree to become verifiers, although they will defect in some way as discussed below. Data owners wrongly depend on actively selfish peers contrary to passively selfish ones. Such a behavior constitutes an attack to the cooperation scheme much more offensive because it wastes network bandwidth and lowers data reliability and trust towards verifiers.

Active selfishness involves addressing the following issues:

– *Resource related collusions:* peers may collude to reduce their resource usage (storage space, computation, or bandwidth utilization). Holders may destroy all replicas of a data but one and still be able to answer verification requests. This attack might be addressed by personalizing data replicas at each holder ([4], [18]). Another type of collusion may occur between a holder and its verifier such that neither the holder stores data nor the verifier performs verification operations. This can be avoided by distributing the verification task to several verifiers, then checking the consistency of their answers to decide on the trustworthiness of both the holder and the verifiers. If verifiers might defect altogether (Quis custodiet ipsos custodes[1]), the owner might ultimately perform sampling verifications itself, which should be done sparingly, for obvious resource related matters.
– *Remuneration related vulnerabilities:* attacks may directly target the incentive mechanism to defeat its filtering capabilities. The Sybil attack represents another potential vulnerability making it possible to generate new peers at will. The Sybil attacker masquerades under multiple simultaneous peer identities in order to gain an unfair advantage over system resources. The payment based mechanisms to be envisioned should therefore support some form of real world based authentication. This attack should at least be mitigated by imposing a real world monetary counterpart to membership for peers joining the storage system. In terms of implementation, the payment mechanism should also prevent or mitigate in some respect double spending attacks.

Purely malicious behaviors are also possible: a malicious peer might aim at obtaining or altering some private data, or at destroying the data stored or the infrastructure through denial of service attacks (e.g., flooding), even at the expense of its own resources. This paper only focuses on selfish behaviors, in particular active ones, and their mitigation through a (micro)payment based incentive scheme.

3 Related Work

A number of micropayment schemes have been proposed in the past like PayWord, MicroMint [20], and Millicent [21]. Those rely on a central broker, the

[1] Who watches the watchers?

bank, that tracks each peer balance and payment transactions. In most of these schemes, the load of the bank grows linearly with the number of transactions; though hash chains in PayWord or the use of electronic lottery tickets [23] greatly reduce such cost. As example, the P2P micropayment system MojoNation [19] has also a linear broker's load and this system has gone out of work because the central bank constitutes as well a single point of failure.

One obvious way of distributing bank's work would be to use super-peers disseminated within the P2P network. These super-peers would provide neutral platforms for performing a payment protocol. The use of such an infrastructure of trusted peers may make sense, in particular in relation with content distribution networks (CDNs) [3]. Such networks involve the deployment of managed workstations all over the Internet, thereby providing a nice platform for other functionalities.

The scale of the P2P system makes it necessary to resort to a type of protocols termed optimistic protocols where the bank does not necessarily take part in the payment, but may be contacted to arbitrate litigations between peers. With such type of protocols, the bank's work is reduced. PPay [22] is a lightweight micropayment scheme for P2P systems where the issuer of any coin is a peer from the system that is responsible for keeping trace of the coin. However, the bank comes into play when the issuer of a coin is off-line. In a very dynamic system, the probability of finding the original issuer of the coin on-line is very low. In this situation, PPay converges to a system with a centralized bank.

If data storage should be achieved in a large-scale and open P2P system, designs based on a trusted authority may be unfeasible or unmanageable. In that case, implementing the optimistic fair exchange protocol would have to be done by relying solely on peers. [7] describes design rules for such cryptographic protocols making it possible to implement appropriate fair-exchange protocols.

To the best of our knowledge, the only fully-decentralized micropayment scheme that has been proposed so far is KARMA [14]. KARMA splits the bank functionality in different bank sets composed of peers selected and appointed randomly from the P2P system for each peer when it first joins the system. The KARMA payment scheme does not require any trusted infrastructure and is scalable.

The payment scheme proposed in this paper relies on such framework to administer peer accounts. Enforcing the fair-exchange of payment against a cooperative service is based on a cryptographic protocol (described in Section 4) that is built on top of the KARMA framework. The framework has been initially applied to the file sharing problem described in [14]. The described application cannot be assimilated to a P2P storage application since in the former case payments are immediately charged after the exchange of the file, whereas in the latter case payments for storage or verification are by installment i.e., they are billed at a due date that corresponds to the confirmation (by verifications) of the good behavior of the holder or the verifier. Therefore, we will supplement the KARMA framework by an escrowing mechanism (described in detail in

Section 4) that guarantees the effective payment of credits promised by the owner towards a cooperative holder or a cooperative verifier.

Tamper resistant hardware (TRH) have also been suggested as a way to enforce payment protocols in a decentralized fashion as illustrated by smart cards in [6]. TRH supported approaches have been suggested within the TermiNodes [1] and CASHnet [2] projects as a support for implementing cooperation incentives. TRH based approaches suffer from other attacks on the payment scheme: if the TRH of a non cooperative or malicious peer is disconnected from the other peers, their credits/tokens might not be available, which might raise starvation issues. However, the use of a secure operating system as a TRH might make it possible alleviate this problem notably by more completely controlling and possibly reducing the device functionalities if the peer does not connect to the system network.

4 Payment-Based Storage

In this section, we first give an overview of the payment-based storage scheme, to describe then the cryptographic protocol that achieves such scheme.

4.1 Overview

We propose a mechanism that monitors data storage on a regular basis to determine the payments between data owners, holders, and verifiers. The payment mechanism allows a peer storing data for other peers to be paid for its service. It thus controls the storage functions seen in the previous section by rewarding cooperating peers.

Notations: Let B_P denote the bank-set of the peer P, PK_P the public key of a peer P, SK_P the private key of P, and SK_{B_P} the private key of the bank-set of P. A message M signed by some key K is denoted as M_K or σ_K (bank-set signature is explained in [14]).

Let G denote a finite cyclic group with n elements. We assume that the group is written multiplicatively. Let g be a generator of G. If h is an element of G then finding a solution x (whenever it exists) of the equation $g^x = h$ is called the discrete logarithm problem (DLP) and is assumed hard to solve.

Assumptions: A P2P system generally consists of altruistic peers, selfish peers, malicious peers, and others with behavior ranging in between. We will assume that there are a non-negligible percentage of the peers that are altruistic or at least correctly follow the protocol. Peers of the storage system are structured in a DHT such as CAN [8], Chord [10], Pastry [9], or Tapestry [11]. A DHT consists of a number of peers having each a key $Key(Peer)$ in the DHT space, which is the set of all binary strings of some fixed length. Each participant is assigned a secure, random identifier in the DHT identifier space: $Id(Peer)$. We assume that the DHT provides a secure lookup service (see [12] and [13]): a peer supplies an arbitrary key (an element in the DHT space), and the lookup service returns the active node in the DHT that stores the object with the key.

Peer selection: To avoid pre-set collusion between verifiers and the owner or verifiers and the holders, holders and verifiers should be randomly chosen. The random selection of peers is generally used for its simplicity since it is less sophisticated and since it consumes less bandwidth per peer. TotalRecall [24] and the P2P storage system of [25] rely on a DHT to randomly select data holders. The selection is realized by randomly choosing a value from the DHT address space and routing to that value. [26] proved the positive effects of randomization in peer selection strategies. For instance, a long-list of randomly chosen potential holders and verifiers can be constructed: the l_1 (respectively l_2) closest peers in the DHT identifier space to the key $HASH_H(Id(Owner), timestamp)$ (respectively $HASH_V(Id(Owner), timestamp)$) constitute the potential holders (respectively verifiers) of the owner ($HASH_H$ and $HASH_V$ are pseudo-random functions publicly known).

Credit escrowing: Each peer has a personal account managed by a set of peers likewise KARMA [14] that are called bank-set. Our payment scheme relies on digital checks. To prevent peers from emitting bad checks, the amount of credits that corresponds to a check value are escrowed, i.e., the necessary number of credits to pay check holder are locked by the bank-set. Consequently, bank-sets keep two types of peer balances: normal credits and locked credits. Credits are escrowed for some time-out (that corresponds to the check's expiry time), after which they are returned to the peer normal balance. The owner desiring to store data in the system must be able to pay its holders and verifiers with checks. That's why, it must escrow credits which are converted to digital checks. These checks are then stored in a blinded version at the corresponding holders and verifiers. Checks include some random numbers that are generated by the owner and certified by its bank-set. The latter have a blinded version of these numbers too (to prevent collusion between one bank-set member and a holder or a verifier). Each blinded digital check has this form:

$$C(payer, payee, g^c) = \{g^c, id(payer), id(payee), price, seq, TTL, \sigma_{SK_{payer}}\} \quad (1)$$

having c a random number, seq check's sequence number, and TTL check's expiry time. The knowledge of c by the payee allows this latter to be paid credits of value price. The bank-set of the payer is not informed of this number c; but only a blinded version of it g^c. The verification operation allows both the verifier and the holder to extract the check in order to be able to present it to their bank-set to be paid in return. The holder must also escrow an amount of credits corresponding to the punishment it gets if it destroys data that it has promised to store. The escrowed credits of the holder are converted to one digital check that is certified by the holder's bank-set. The check is split into multiple shares each one will be stored at each verifier: a threshold number k of these shares allow reconstructing the full check. Shares of the digital check comprise the following numbers (in blinded version) $\{g^{s_i}\}_{1 \leq i \leq n_v}$ which are shares of g^s if $\{s_i\}_{1 \leq i \leq n_v}$ are shares of s [16]. If a threshold-based majority of verifiers agree that the holder has destroyed data, they can construct holder's check and present it to the owner such that this latter will be reimbursed.

Data verification: Each verifier appointed by the owner periodically checks storage of data stored at a holder. The verifier does not have the full challenge for the holder, but rather a share of the challenge: a threshold number of messages received by the holder from verifiers allow this latter to reconstruct the full challenge. Distributing the verification task to multiple verifiers prevents potential collusion between the holder and a verifier, since the holder cannot compute the response without the stored data even by having a partial knowledge of pre-computed challenges (metadata) disclosed by a small number of colluding verifiers. The verification operation has three-fold objectives: it allows assessing the availability of stored data, it permits the verifier to remove the blinding factor of the stored digital check in order to get paid for verification, and finally it allows the holder to recover also its check for its payment too. Since, the verifier is paid exactly for each verification operation it actually performs, verification operations are executed in a defined number. Consequently, the payment scheme does not require a verification protocol where verifications are unlimited and may rely on pre-computed challenges for instance.

The originality of the scheme mainly stems from the combination of verification and payment operations such that the scheme works without the data owners being involved. These latter can store their data at multiple holders, appoint verifiers, and then they can forget about the stored data until of course the moment of retrieval. The system of peers operates automatically without the intervention of owners: it periodically checks data storage and pays the cooperating holders.

4.2 Protocol

In this section, we describe a protocol that provides a cryptographic implementation of the scheme. We consider an owner denoted O that stores its data at a holder H. The integrity of such data is periodically checked by a verifier V on behalf of O. The proposed protocol consists in multiple steps described in the following Fig. 1.

5 Simulation Experiments

In order to validate the ability of our payment-based storage approach to detect and punish selfish peers, we developed a custom simulator of our payment scheme. This section first describes the framework of simulations, then presents and analyzes the results obtained.

5.1 Simulation Framework

The self-organizing storage system is modeled as a closed set of homogeneous peers. The storage system operation is modeled as a cycle-based simulation. One simulation cycle corresponds to the period between two successive verifications.

Operations	**Description**
Credit escrowing	**(*O* escrows credits for the payment of *H* and *V*)** *O*: fix number of verification operations to *m* *O*: generate random numbers $\{R_i\}_{1\leq i\leq m}$, v_H, v_V *O*: compute for each $i\in[1, m]$ $T_i = HASH(HASH(d, R_i), v_H)$ $T'_i = HASH(HASH(d, R_i), v_V)$ $O \rightarrow B_O$: $\{g^{T_i}, price\}_{1\leq i\leq m}, id(H), \{g^{T_i}, price\}_{1\leq i\leq m}, id(V)$ $B_O \rightarrow O$: $\{C(O, H, g^{T_i})\}_{1\leq i\leq m}$, $\{C(O, V, g^{T'_i})\}_{1\leq i\leq m}$ **(*H* escrows credits to form its punition *p*)** *H*: generate a random number *s* *H*: generate $\{s_i\}_{1\leq i\leq n_v}$ shares of *s* $H \rightarrow B_H$: $\{g^{s_i}\}_{1\leq i\leq n_v}$, g^s, $id(O)$, *punition* B_H: check $\{g^{s_i}\}_{1\leq i\leq n_v}$ are shares of g^s $B_H \rightarrow H$: $\{C(H, O, g^{s_i})\}_{1\leq i\leq n_v}$
Data storage	**(*O* stores data *d* at *H*)** $H \rightarrow V$: $s_i, C(H, O, g^{s_i})$ $V \rightarrow H \rightarrow O$: $\{ACK\}_{SK_V}$ $O \rightarrow H$: d, $\{C(O, H, g^{T_i})\}_{1\leq i\leq m}$, v_H **(*O* delegates verification of *d* to *V*)** *O*: generate for each $i\in[1, m]$ $\{r_{ij}\}_{1\leq j\leq n_v}$ shares of R_i *O*: compute $\{HASH^2(d, R_i)\}_{1\leq i\leq m}$ ($HASH^2$: *HASH* is executed 2 times) $O \rightarrow V$: $\{HASH^2(d, R_i)\}_{1\leq i\leq m}$, r_{ij}, $\{C(O, V, g^{T'_i})\}_{1\leq i\leq m}$, v_V
Data verification	**(*V* sends a share of the *i*th challenge to *H*)** $V \rightarrow H$: i, r_{ij} **(*H* answers verifiers upon construction of challenge from shares)** *H*: compute Res=$HASH(d, R_i)$ $H \rightarrow V$: *Res* **(*V* checks *H*'s answer)** *V*: check $HASH(Res) =?$ $HASH^2(d, R_i)$
Payment	**(*H* obtains its *i*th payment)** *H*: compute $T_i = HASH(HASH(d, R_i), v_H)$ $H \rightarrow B_H$: T_i, $C(O, H, g^{T_i})$ $B_H \rightarrow B_O$: T_i, $C(O, H, g^{T_i})$ B_H: increase *H*'s balance B_O: decrease *O*'s balance **(*V* obtains its *i*th payment)** *V*: compute $T'_i = HASH(HASH(d, R_i), v_V)$ $V \rightarrow B_V$: T'_i, $C(O, V, g^{T'_i})$ $B_V \rightarrow B_O$: T'_i, $C(O, V, g^{T'_i})$ B_V: increase *V*'s balance B_O: decrease *O*'s balance
Data retrieval	**(*O* retrieves *d* from *H*)** $H \rightarrow O$: d **(*H* unblocks its escrowed credits)** $O \rightarrow H \rightarrow B_H$: $\{ACK\}_{SK_O}$ If *TTL* times out: unspent escrowed credits are returned (respectively to *O* and *H*)

Fig. 1. Payment protocol

Churn: Peers arrive to the system in Poisson distribution: there are 100 newcomers per hour, for an average lifetime of 2 weeks. [17] shows that Gnutella peer uptime follows a power-law distribution. We will use the same distribution for peer uptime and downtime. In average, a peer stays online for 1 hour and connects in average 6.4 times in a day.

Storage: Peer storage space, file size, and storage duration are chosen from truncated log-normal distributions. The storage space of each peer is chosen from 1 to 100GB, with an average of 10GB. In each day of simulated time, 2.85 of files are stored per peer for an average period of 1 week. The average file size is 500MB. The stored files will be checked by verifiers each day.

User strategies: We consider three peer strategies: cooperation, passive selfishness (free-riding) and active selfishness.

- **Cooperative:** Whenever the peer accepts to store data from another peer, it keeps them stored. Whenever the peer accepts to check the availability of some data at a storage peer, it will periodically perform verification operations on this peer as agreed.
- **Passively selfish:** The peer will never accept to store data and will never accept to verify the availability of some data stored for other peers. The peer is just consumer of the storage system. This type of behavior is also termed free-riding.
- **Actively selfish:** The peer probabilistically accepts to store data for other peers or to verify storage at other peers. Whenever it stores or verifiers for others, it will fulfill its promise only probabilistically. This type of behavior has an instability effect: alternating between cooperation and selfishness at a given probability: probability of participation denoted p and probability of achieving promise denoted q.

5.2 Simulation Results

Different scenarios were simulated to analyze the impact of several parameters on the payment mechanism. Simulation studies the transition phase of the network to a stable state where cooperative peers are the only active actors of the system. Used notations are summarized in Table 1.

Exclusion of selfish owners: Fig. 2. demonstrates that selfish peers have less capability over time to store data in the system; on the other hand, cooperative peers are becoming the majority of data owners in the storage system. Passive selfish peers are the first to be excluded from the system because they consume all their initial credits (all peers have a default number of credits when they join, in order to facilitate system bootstrap). Active selfish peers are also filtered out from the system because they cooperate only probabilistically. The figure shows also that a decreasing fraction of these active selfish peers are still present in the system. Because they cooperate at some probability; they may temporarily gain some credits and then go without detection. These are considered as the false

Table 1. Description of notations used in simulation figures

Notations	Description
n	Number of peers in the system
r	Data replication factor (number of holders of some given data)
m	Number of verifiers of some given data
p	Probability of participation of an actively selfish peers
q	Probability of data conservation by an actively selfish peer
w	Weight parameter in price formulation (associated with the amount of credits owned by a given peer)

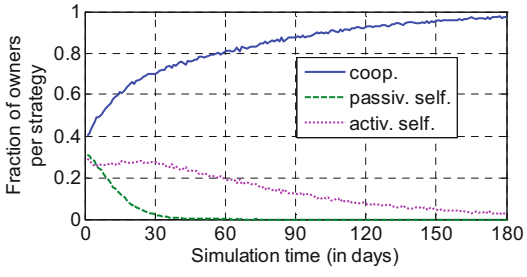


Fig. 2. Averaged ratio of owners per strategy. n=1000, r=3, m=5, w=0.5, p=0.2, q=0.2, 40% cooperators, 30% passively selfish peers, 30% actively selfish peers.

negatives of our detection scheme. But still, such false negatives are decreasing with time. We may notice that 1 simulated month is sufficient to filter out passively selfish peers; however the filtering may take more than 3 months for actively selfish peers. Yet, this time period can be reduced by adaptively reducing the default initial income for newcomers.

Exclusion of selfish holders: Fig. 3. depicts the fraction of cooperators and selfish peers in the population of data holders. The figure demonstrates that with time cooperative peers will make the majority of holders. This result is due to the fact that actively selfish peers are losing their credits and then becoming unable to escrow credits necessary for the storage of other peers' data; albeit the fact that they will propose small prices (this explains the small pick in the first simulated month).

Overhead: Note that we only measure the communication overhead due to holder and verifier selection and storage verification. In particular, we exclude the cost of P2P overlay maintenance and storing/fetching of files, since it is not relevant to our analysis. In a further observation, the bandwidth consumed for verification is dependent on the number, rather than the size, of files being stored. This is in fact a requirement on the verification protocol. Fig. 4 shows the amount of control messages per file. The figure demonstrates that the bandwidth cost decreases with time.

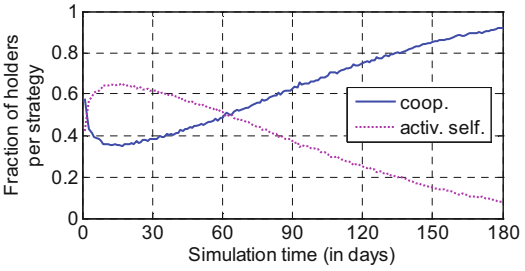


Fig. 3. Averaged ratio of holders per strategy. n=1000, r=3, m=5, w=0.5, p=0.2, q=0.2, 40% cooperators, 30% passively selfish peers, 30% actively selfish peers.

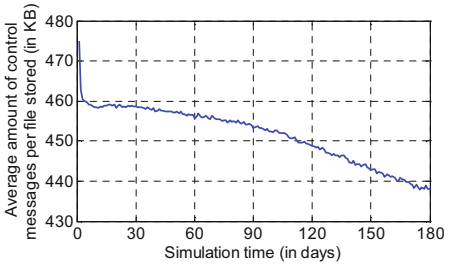


Fig. 4. Average amount of control messages per file stored (in KB). n=1000, r=3, m=5, w=0.5, p=0.2, q=0.2, 40% cooperators, 30% passively selfish peers, 30% actively selfish peers.

Data reliability: Fig. 5 shows that the rate of the amount of data injected into the storage system decreasing. This is due to several factors. First of all, there is the gradual exclusion of selfish peers that limits the number of peers able to store data in the system. Second, there are possible false positives of our detection system due to the starvation phenomenon where cooperative peers are not able to contribute because they are not chosen as holders or verifiers, and at the end they consume all their credits and get expelled from the system. The figure also depicts the rate of file loss that is falling down as low as zero, owing to the exclusion of selfish holders (explained earlier on).

5.3 Security Considerations

In this section, we analyze the security of the protocol to prevent or at least mitigate the threats described in section 2. The security of our scheme relies principally on replication to deter peers that might try to subvert the protocol. It assumes that there are at least a given number of peers in the system at all times, and uses protocols to ensure that the system will correctly operate unless a substantial fraction of these peers are selfish or malicious.

Selfishness punishment: The proposed payment scheme works as a quota system: peers have to keep a given balance to be able to participate to the storage system.

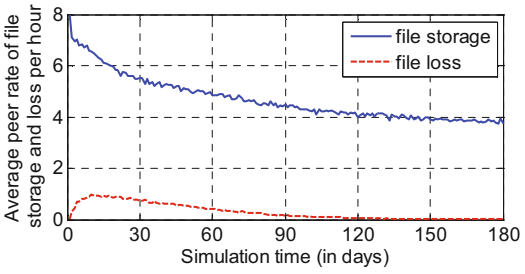


Fig. 5. Average peer rate of file storage and loss per hour. n=1000, r=3, m=5, w=0.5, p=0.2, q=0.2, 40% cooperators, 30% passively selfish peers, 30% actively selfish peers.

Peers that are passively selfish gradually consume all their credits for their data storage and when their accounts are exhausted they will not be able to use the storage system anymore. In the same way, actively selfish peers keep losing credits because they have been detected destroying data they have promised to store. These peers will also drain their accounts and with time will not able to use the storage system.

Collusion prevention: Holder and verifier selection is random which limits preset peer collusions. The digital check of the holder is shared among verifiers, thus mitigating also collusion between the owner and one or a small number of verifiers. Additionally, challenges sent to the holder are constructed cooperatively by verifiers to avoid collusion between the holder and one or a small number of verifiers. Finally, collusion between the owner and the holder is less probable because it does not generate any financial profit since the owner must pay verifiers to check holder's storage. The distribution of tasks to several verifiers limits collusion; but it is still feasible if at least k verifiers collude with the holder for instance. The probability of collusion can be computed as:

$$\sum_{i=k}^{n_v} \binom{n_v}{i} p^i (1-p)^{(n_v - i)} \tag{2}$$

where p is the probability that a given verifier is not honest (colluder). However such collusion probability is less than 0.1 for 60% of dishonest peers (i.e., $p = 0.6$) in the system and with $n_v = 10$ and $k = 8$, or 80% of dishonest peers and with $n_v = 20$ and $k = 18$.

Fair-exchange: Holder and verifier payments are strongly related to the correct operation of data verification. This motivates the holder to accept storage verifications and incites verifiers to perform this task periodically on behalf of the owner. The frequency of verifications is determined by the owner at the time of delegating verification. This frequency is the matter of all verifiers: the majority of verifiers use the determined frequency at which the holder collects a sufficient number of random numbers to compute the challenge; therefore very fast or very slow frequencies of some verifiers do not influence (with large probability) the actual frequency of computing the verification challenge. The holder or the

verifier cannot cash their checks without verifying the stored data. This is due to two reasons. First of all, the secret random numbers included in the checks are only known to the actual payers since they are held in DLP-based blinded version at the payees and bank-sets too. Second, $HASH$ is a one-way function, so knowing $HASH$(m) does not give any extra information on m. Therefore, the data verification operation strongly relates to the holder and the verifier payment operations. However, the existence of such relation is only guaranteed by the owner. So, if a verifier or a holder is still not paid even though it behaves well, it has the possibility to prove owner's misbehavior to the other participants (using the certified checks) and also to stop cooperating with the owner without being punished. Thus, the owner is encouraged to provide this type of relation to secure the future cooperation of peers handling its data. The bank-set comes into play to guarantee that payments are actually doable since the corresponding amounts of credits are locked to prohibit the payer from emitting bad checks.

Remuneration-related attacks: Attacks on the payment scheme (such as double spending or impersonation) are handled by the KARMA framework. Moreover, the sequence number and the identity of the payee included in each payment receipt prevent replay attacks, because they impose that the digital check is only cashed by the payee one time. We assume that all exchanged messages are signed and encrypted by the keys of the involved parties in order to ensure the integrity of exchanged messages and even the security against man-in-the-middle attack for instance. Sybil attacks are mitigated a la KARMA by compelling peers to execute a cryptographic puzzle before joining the storage system, the result of which will be used to construct their identities.

6 Conclusion

We described a new payment scheme as an incentive for P2P storage. The scheme is scalable and cost-effective since it does not require any trusted infrastructure. It relies on a cryptographic protocol to enforce the fair exchange of payments against cooperative behavior. The protocol is self-powered: it does not require the mediation of data owners during the data verification process or for payments. Additionally, the design of the scheme takes into consideration several types of peer collusions, and achieves an effective punishment of selfishness. Our simulations outline that this scheme makes the system converge to a stable state where non-cooperative peers are finally filtered out. We are further investigating the formal validation of this approach.

References

1. Buttyán, L., Hubaux, J.-P.: Nuglets: a virtual currency to stimulate cooperation in self-organized ad hoc networks. Technical report, EPFL (2001)
2. Weyland, A., Staub, T., Braun, T.: Comparison of Incentive-based Cooperation Strategies for Hybrid Networks. In: Braun, T., Carle, G., Koucheryavy, Y., Tsaoussidis, V. (eds.) WWIC 2005. LNCS, vol. 3510, pp. 169–180. Springer, Heidelberg (2005) ISBN: 3-540-25899-X

3. Akamai technologies, Inc., http://www.akamai.com/
4. Oualha, N., Nen, M., Roudier, Y.: A Security Protocol for Self-Organizing Data Storage. In: 23rd International Information Security Conference (SEC 2008), Milan, Italy (September 2008)
5. Oualha, N., Nen, M., Roudier, Y.: A Security Protocol for Self-Organizing Data Storage. Technical Report N RR-08-208, EURECOM (January 2008) (extended version)
6. Vogt, H., Pagnia, H., Gärtner, F.C.: Using Smart Cards for Fair Exchange. In: Fiege, L., Mühl, G., Wilhelm, U.G. (eds.) WELCOM 2001. LNCS, vol. 2232, pp. 101–113. Springer, Heidelberg (2001)
7. Asokan, N., Shoup, V., Waidner, M.: Asynchronous protocols for optimistic fair exchange. In: Proceeding of the IEEE Symposium on Security and Privacy, Oakland, CA, USA, May 3-6, pp. 86–99 (1998)
8. Ratnasamy, S., Francis, P., Handley, M., Karp, R., Shenker, S.: A scalable content-addressable network. In: Proceedings of SIGCOMM, San Diego, CA, August 27-31 (2001)
9. Rowstron, A., Druschel, P.: Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In: Liu, H. (ed.) Middleware 2001. LNCS, vol. 2218, p. 329. Springer, Heidelberg (2001)
10. Stoica, I., Morris, R., Karger, D., Kaashoek, M.F., Balakrishnan, H.: Chord: A scalable peer-to-peer lookup service for internet applications. In: Proceedings of SIGCOMM, San Diego, CA, August 27-31 (2001)
11. Zhao, B.Y., Kubiatowicz, J., Joseph, A.D.: Tapestry: An infrastructure for fault-tolerant wide-area location and routing. Technical Report UCB//CSD-01-1141, University of California, Berkeley (April 2000)
12. Sit, E., Morris, R.: Security Considerations for P2P Distributed Hash Tables. In: Druschel, P., Kaashoek, M.F., Rowstron, A. (eds.) IPTPS 2002. LNCS, vol. 2429, p. 261. Springer, Heidelberg (2002)
13. Castro, M., Druschel, P., Ganesh, A., Rowstron, A., Wallach, D.S.: Secure routing for structured peer-to-peer overlay networks. In: Symposium on Operating Systems and Implementation, OSDI 2002, Boston, MA (December 2002)
14. Vishnumurthy, V., Chandrakumar, S., Sirer, E.G.: KARMA: A Secure Economic Framework for P2P Resource Sharing. In: Proceedings of the Workshop on the Economics of Peer-to-Peer Systems, Berkeley, California (June 2003)
15. Ray, I., Ray, I.: Fair Exchange in E-Commerce. ACM SIGEcomm Exchange 3(2), 9–17 (Spring 2002)
16. Desmedt, Y.G., Frankel, Y.: Threshold Cryptosystems. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 307–315. Springer, Heidelberg (1990)
17. Stutzbach, D., Rejaie, R.: Towards a Better Understanding of Churn in Peer-to-Peer Networks. Technical Report CIS-TR-04-06, University of Oregon (November 2004)
18. Lillibridge, M., Elnikety, S., Birrell, A., Burrows, M., Isard, M.: A Cooperative Internet Backup Scheme. In: Proceedings of the 2003 Usenix Annual Technical Conference (General Track), San Antonio, Texas, pp. 29–41 (June 2003)
19. MojoNation archived website, http://web.archive.org/web/20020122164402/%20,http://mojonation.com/
20. Rivest, R.L., Shamir, A.: Payword and micromint: two simple micropayment schemes. In: Lomas, M. (ed.) Security Protocols 1996. LNCS, vol. 1189. Springer, Heidelberg (1997)
21. Glassman, S., Manasse, M., Abadi, M., Gauthier, P., Sobalvarro, P.: The millicent protocol for inexpensive electronic commerce. In: Proceeding of WWW4 (1995)

22. Yang, B., Garcia-Molina, H.: PPay: Micropayments for Peer-to-Peer Systems. In: ACM Conference on Computer and Communications Security (CCS 2003), Washington, DC, USA (October 2003)
23. Rivest, R.L.: Electronic lottery tickets as micropayments. In: Luby, M., Rolim, J.D.P., Serna, M. (eds.) FC 1997. LNCS, vol. 1318, pp. 307–314. Springer, Heidelberg (1997)
24. Bhagwan, R., Tati, K., Cheng, Y.-C., Savage, S., Voelker, G.M.: TotalRecall: System Support for Automated Availability Management. In: ACM/USENIX NSDI (2004)
25. Oualha, N., Roudier, Y.: Reputation and Audits for Self-Organizing Storage. In: The 1st Workshop on Security in Opportunistic and SOCial Networks (SOSOC 2008), Istanbul, Turkey (September 2008)
26. Brighten Godfrey, P., Shenker, S., Stoica, I.: Minimizing churn in distributed systems. ACM SIGCOMM CCR 36(4) (2006)
27. Deswarte, Y., Quisquater, J.-J., Sadane, A.: Remote Integrity Checking. In: Proceedings of Sixth Working Conference on Integrity and Internal Control in Information Systems, IICIS (2004)

STARS: A Simple and Efficient *S*cheme for Providing Transparent *T*raceability and *A*nonymity to *R*eputation *S*ystems

Zonghua Zhang[1,⋆], Jingwei Liu[2], and Youki Kadobayashi[3]

[1] IT/TELECOM Lille 1, France
[2] Xidian University, China
[3] iSRC, NICT, Japan

Abstract. Reputation systems play a vital role in constructing mutual trust relationships between different entities in autonomic computing networks by enforcing them to act as prescribed protocols or specifications. They can be, however, subverted and abused if the association rules between an entity's identity and its reputation is exploited. While various anonymizating techniques can be used to prevent that, the extent of anonymity is extremely hard to be determined at an appropriate level, potentially allowing sophisticated attackers to correlate a party with its reputation. To manifest and further gain insights into such vulnerabilities, we systematically decompose the reputation system into four components from a functional perspective and use a set of performance metrics to examine them. Specifically, a new attack taxonomy is given, and a simple scheme termed STARS, which is transparent to particular reputation systems, is proposed for achieving both anonymity and traceability. We finally discuss implementation issues and validate performance through case studies, comparative analysis, and simulations.

1 Introduction

Reputation system, which has gained widespread application in peer to peer networks [27], commercial online communities [9], and mobile ad hoc networks [2, 4, 11], potentially plays a vital role in building mutual trust relationships between different entities in autonomic computing networks by enforcing them to truthfully act as specified protocols. Relying on a reputation system, each entity rates the behavior of its peers, a reputation score of a particular entity is derived from all the ratings it gains, serving as a reference to decide its future interactions with the others, thereby encouraging normal behavior and punishing abnormal ones.

It is believed that identity management, which recognizes network entities and associates their behaviors with corresponding reputation, is essential to reputation system. To preserve privacy of interest, some anonymizating techniques can be used to conceal the real identities of network nodes and de-associate node

⋆ Most of this work was done when the author was with iSRC of NICT, Japan.

J. Garcia-Alfaro et al. (Eds.): DPM 2010 and SETOP 2010, LNCS 6514, pp. 170–187, 2011.

identities with reputation scores. This can be practically implemented by existing anonymous credentials that relies on pseudonymization. But one conflict may arise: the more persistent of node identity the more accurate a reputation system can rate nodes and the more reliable of reputation calculation, but meanwhile it is even easier for an attacker to infer the linkability between node identity and reputation [19, 20]. On the contrary, frequent pseudonym updates may lead to extra overhead and foster such attacks as white-washing and sybil. More seriously, a sophisticated attacker can associate the changes of pseudonym of a particular node with its reputation updates and further trace back the historic behavior [1].

This indicates that an implicit trade-off exists between node anonymity and reputation, and it must be examined and balanced to an appropriate extent for attaining secure, dependable, and effective reputation management. To that end, a number of challenges must be tackled: how the node behavior is associated with its reputation while preserving anonymity; to what extent node anonymity should be maintained so that appropriate trade-offs between different operational metrics can be achieved; how to trace back malicious attackers (or traitors) in Anonymous Reputation Systems (ARS) for identifying malicious nodes.

Despite the research effort on the first challenge, the latter two are comparatively neglected. In this paper, we systematically decompose reputation systems into four functional components and propose a set of operational metrics for performance evaluation. Then the major operational vulnerabilities and their countermeasures are examined. As a result, a novel scheme termed STARS is developed as a plug-in for providing transparent anonymity and traceability to reputation systems.

2 Related Work

Reputation systems have been widely applied to various computing environments [13, 14, 32], but they are often treated as an intact system and lack of an insightful analysis on the operational metrics. This paper generally decomposes reputation systems into four functional components and proposes a set of key metrics for examining their operational characteristics, manifesting system vulnerabilities. Usually the design and implementation of reputation systems are tailored for particular contexts, which may adopt security mechanisms like authentication, authorization, and cryptographic routing protocols, contributing to secure reputation management [6, 11, 19, 25]. However, anonymity has never been treated as an independent property, and the implicit relationship between pseudonymity and reputation has not been identified in a fine-grained way for manifesting the potential vulnerabilities.

The privacy-preserving reputation management was discussed in [26], and another work was reported in [21], which particularly considers sybil attacks and excludes other vulnerable operations and operational metrics. Recently a new cryptographic primitive called *signatures of reputation* was proposed in [3]

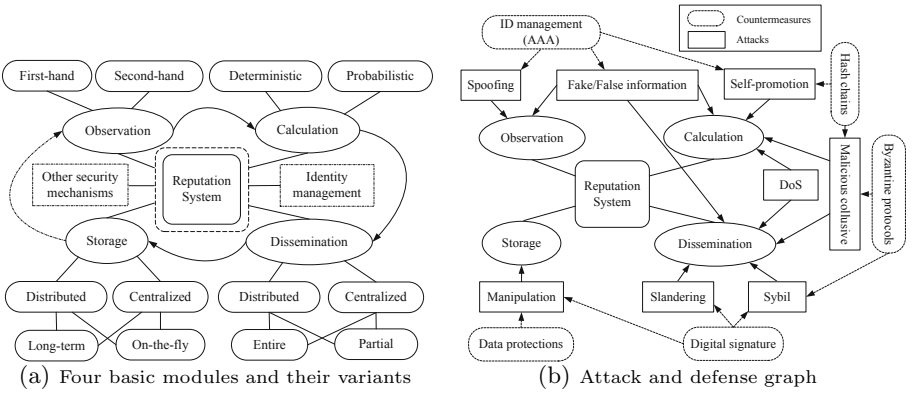
for supporting monotonic measures of reputation while keeping anonymous. But this scheme is built from the scratch and cannot be generally applied to other reputation systems.

Along with anonymity, traceability is another desirable property, especially in such application scenarios as virtual organizations [15]. Sharing a similar goal, our scheme is more tailored for general reputation systems, and the newly designed cryptographic protocols make it more applicable. Also, while various attacker behavior in plain reputation systems have been discussed [13], to the best of our knowledge, insider attacks on ARS have not been sufficiently examined. We believe traitors in ARS pose much more threat in that they have more knowledge regarding system vulnerabilities security mechanisms.

3 Reputation System: Operational Metrics, Vulnerabilities, and Attacks

From a functional perspective, a reputation system can be decomposed into four components, as shown in Fig. 1(a), implying four major operations:

- *Observation collection*: first-hand observation are collected by direct interactions between participants, while the second-hand ones are indirectly recommended.
- *Calculation*: either probabilistically or deterministically creates, maintains, and updates reputation scores.
- *Dissemination*: either initiated periodically by some master nodes (active mode) or on the request of some applications (passive mode).
- *Storage*: either decentralized or centralized; the reputation scores are either stored in a complete form (long-term maintained) or calculated in real-time (on-the-fly).



(a) Four basic modules and their variants (b) Attack and defense graph

Fig. 1. Reputation system, its components, vulnerabilities, and countermeasures

We thus have the following observations: (1) the four modules can be selectively integrated together for composing the most appropriate reputation system as different application scenarios; (2) for a particular scenario, the trade-offs between a set of operational metrics can be balanced for achieving the best; (3) the operations must be seamlessly integrated with those underlying security mechanisms, avoiding any negative consequence. Our design takes into account these observations.

3.1 Operational Metrics

Clearly a set of operational metrics is necessary for examining the quality and soundness of reputation systems, as well as their integration with other security counterparts and identity management schemes. A number of investigations on the existing reputation systems [13, 14, 32] facilitate us to figure out the following ones, which are intended to cover all the factors associated with those major operations indicated in Fig. 1(a).

- *Computational overhead*—includes computational complexity of reputation calculation, storage overhead, and communication cost for reputation dissemination.
- *Accuracy*—implies that any slight deviation from the anticipated behavior can be captured and reflected by reputation values.
- *Scalability*—the performance can be kept at an acceptable level in the face of changing network scale, which is mainly related to reputation dissemination and storage.
- *Adaptability*—adapts to the changing network topology while preserving acceptable performance, dominated by all the three operational modules.
- *Sensitivity*—deviation of node behavior can be detected and reported within minimal delay.
- *Security and Privacy*—preserves integrity, confidentiality and non-repudiation of reputation values, as well as such node privacy as actual identity and trust ratings.
- *Robustness*—the performance does not dramatically deteriorate in the presence of accidental system errors like communication link interruption, traffic congestions.

Moreover, *traceability* enables a trusted party to disclose the anomalous behavior of actual network nodes for maintaining ARS performance and solving legislation issues.

3.2 Operational Vulnerabilities and Countermeasures

While reputation system vulnerabilities have been extensively studied [13, 18], the analyses usually treat the system as a whole. To have a deep look into the implicit relationships between different operational modules and the potential vulnerabilities, we exemplify a number of typical attacks along with their countermeasures in Fig. 1(b). It is obvious that most of attacks on reputation systems,

spoofing and DoS for example, are not novel, even though the vulnerabilities are operation- and system-specific and can be exploited in various ways. Also, attack surface of reputation system is large and any stage of the operational cycle can introduce vulnerabilities and eventually result in the disruption of the whole system. Accordingly, light-weight cryptographic techniques [33] like asymmetric cryptography, blind signatures [20], hash chain [11], PGP [4], distributed hash table (DHT) [27], and other authorization and authentication techniques can be applied. Also, Byzantine agreement/protocols can be taken as computation basis to make reputation systems robust to false/inaccurate trust value reports [16]. In practice, it is even reasonable to assume the existence of trusted nodes considering the fact that tamper-proof hardware has very small probability of being compromised. The pre-trusted nodes can thus sever as *supervisors* to monitor the whole network by participating in the reputation management [6, 31].

Despite the large variety of vulnerabilities, attack vectors, and countermeasures, our operational analysis reveal three key observations. First, the majority of existing security designs are usually centered on one or more particular operational modules even though they are intended to secure the whole reputation system, thereby leading some hidden vulnerabilities to be remained. Second, a fundamental assumption for the design of reputation systems is that node identity is unique and persistent, which potentially impedes their interaction with anonymous routing protocols. Third, the majority of existing ARS simply assume the existence of some identity authorities relying on public key systems [6, 19–21]. Although sophisticated authentication schemes pertaining to pseudonyms, digital signature, and key management can be applied, their implementation and seamless integration are non-trivial issues. The three observations thus motivate us to take a holistic perspective to examine operational vulnerabilities, subsequently to design more light-weight, in-depth, and effective security mechanisms.

3.3 A New Attack Taxonomy

We generally classify attacks on reputation systems into two categories, which are launched by insider and outsider attackers respectively. In particular, an outsider attacker is usually unauthorized to participate in reputation management and attempts to intercept messages traveling in the reputation system. A successful outsider attack can eavesdrop, spoof, replay or drop messages, consequently subverting reputation systems in terms of availability and integrity. Most of aforementioned attacks fall into this category, which requires more secure and robust authentication and authorization techniques to serve as countermeasures.

Another category of more stealthy and sophisticated attacks is launched by insiders, or we call *traitors*, who are authorized participants in reputation system and have good if not perfect knowledge on the deployed security mechanisms, privileged to participate in any phase of routing/application protocols and further intentionally subvert reputation systems in terms of confidentiality by crafting and manipulating the messages. In particular, no matter what cryptographic primitive is used, a sophisticated insider may manage to identify the implicit relationship between node identity and its reputation, leading to larger

scale attacks and more serious privacy leakage. For instance, an attacker can correlate known information about a network node with the actions performed on behalf of a pseudonym to establish a link between them, and she can even track a sequence of pseudonyms used by the same participant to gather sufficient information for exploring their associations [20]. Furthermore, for an ARS which adopts decentralized dissemination and storage, secret sharing schemes and Byzantine protocols are mostly often used to aggregate trust values. An insider can create malicious collectives to arbitrarily alert the trust ratings assigned to those benign nodes by compromising the aggregation protocols.

4 STARS: A Plug-in to Reputation Systems

This section specifically reports our design, STARS, which servers an add-on to the underlying reputation systems for achieving both anonymity and traceability.

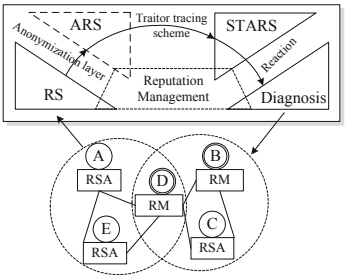
4.1 Design Assumptions and Objectives

We generally assume that reputation systems to be deployed in heterogeneous environments which are equipped with public key cryptographic primitives [30, 33], along with the trusted party like certification authority and pre-trusted reputation manager, which can generate and certify cryptographic keys for various purposes. But we do not necessarily assume the networks to be anonymous [8, 10], while we rather assume anonymity to be an optional property or on-demand service. Also, as Fig. 1(a) indicates, we particularly consider the system operating with distributed dissemination and storage modules, such as the ones deployed in P2P networks and MANETs. We further suppose one or multiple insiders attempting to subvert reputation system.

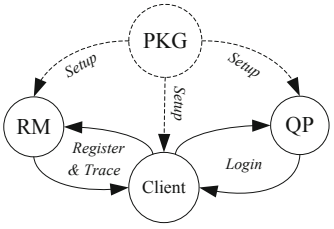
The assumptions allows us to envision such a scheme which can: (1) achieve anonymity in reputation systems regardless of operational environments; (2) enable traceability which allows reputation system manager to (automatically) trace back malicious nodes/agents violating regular operations; (3) serve as a customized plug-in to balance the trade-off between anonymity, traceability, and other operational metrics by taking into account particular operational environments. Finally, the scheme must be light-weight, i.e., its operational cost is negligible.

4.2 Design Architecture and Components

Fig. 2(a) (RS-Reputation System; ARS: Anonymous Reputation System; RSA-Reputation System Agent; RM-Reputation Manager; STARS: the proposed scheme.) illustrates the design architecture, and two typical functional roles are specified: RSA, which locally monitors the neighbors, collects evidence of interest, rates their behavior, and offers trust ratings; RM, which calculates, disseminates, and stores reputation values. For example, A, C, E in Fig. 2(a) can work as RSA while B, D can serve as RM. In practice, the specification of such roles must take into account particular operational environments, and one node



(a) Design architecture: STARS serves as a plug-in for RS/ARS, enabling anonymity and traceability.



(b) Four reputation system entities and their interactive operations in STARS.

Fig. 2. Design architecture and entities of STARS

may play the two roles simultaneously. Since we primarily consider reputation systems in heterogeneous autonomic computing networks, the pre-trusted network nodes can be selected as RM, and both RM and RSA can be mobile, joining or leaving the network freely.

In our design, two other functional roles are introduced: (1) PKG−Public Key Generator, which is usually played by RM and generates cryptographical keys and maintains a partial list of reputation scores for a group of network nodes.; (2) QP−Query Portal, which can be served by either RM or RSA, interacting with the nodes which have reputation requests. In addition, we use *client* to commonly refer to reputation system entities.

4.3 Preliminaries

The design of STARS seeks theoretical foundation from state-of-the-art cryptographic primitives, namely bilinear pairings and Gap Diffie-Hellman Group.

Bilinear Pairings. The bilinear pairings namely Weil pairing and Tate pairing of algebraic curves is defined as a map $e : G_1 \times G_1 \rightarrow G_2$ where G_1 is a cyclic additive group generated by P, whose order is a prime q, and G_2 is a cyclic multiplicative group of the same order q. Let a, b be elements of $\mathbb{Z}_q^*$. We assume that the discrete logarithm problems (DLP) in both G_1 and G_2 are hard. A bilinear pairings has the following properties:

- *Bilinear*: $e(aR, bS) = e(R, S)^{ab}$, $\forall R, S \in G_1$ and $a, b \in \mathbb{Z}_q^*$. This can be related as $\forall R, S, T \in G_1$, $e(R + S, T) = e(R, T)e(S, T)$ and $e(R, S + T) = e(R, S)e(R, T)$;
- *Non-degenerate*: There exists $R, S \in G_1$ such that $e(R, S) \neq I_{G_2}$, where I_{G_2} denotes the identity element of the group G_2;
- *Computable*: There is an efficient algorithm to compute $e(R, S)$ for all R, $S \in G_1$.

Gap Diffie-Hellman Group. Let G_1 be a cyclic additive group generated by P, whose order is a prime q, assume that the inversion and multiplication in G_1 can be computed efficiently. We firstly introduce the following problems in G_1:

- Discrete Logarithm Problem (DLP): Given two elements $R, S \in G_1$, to find an integer $n \in \mathbb{Z}_q^*$, such that $S = nR$ whenever such an integer exists.
- Computation Diffie-Hellman Problem (CDHP): Given P, aP, bP for $a, b \in \mathbb{Z}_q^*$, to compute abP.
- Decision Diffie-Hellman Problem (DDHP): Given P, aP, bP, cP for $a, b, c \in \mathbb{Z}_q^*$, to decide whether $c \equiv ab \bmod q$.

We call G_1 a Gap Diffie-Hellman Group if DDHP can be solved in polynomial time but there is no polynomial time algorithm to solve CDHP or DLP with non-negligible probability. Such group can be found in supersingular elliptic curve or hyperelliptic curve over finite field, and the bilinear pairings can be derived from the Weil or Tate pairings.

4.4 Cryptographic Primitives and Design Principle

STARS employs ID-based signature derived from bilinear pairings on elliptic curves and taking its public identity information as the public key. Formally, four meta algorithms *Setup, Register, Login*, and *Trace* are included, where algorithm *Setup* can be implemented together with reputation systems at deployment stage, algorithm *Register* enables network entities to join in a reputation system by registering to a trusted RM, algorithm *Login* allows RSAs to access reputation system, and the objective of algorithm *Trace* is to identify the nodes violating reputation management polices.

We use Fig. 2(b) to illustrate the operational flow of the four system entities, and the working rationale of STARS can be briefly described as follows: a PKG generates private keys and assigns them to the network nodes (suppose RMs, QPs are pre-selected), which subsequently register to RMs as RSAs by obtaining their unique pseudonyms. During the operation of the reputation system, RSAs can login QPs for querying reputations of the nodes/clients of interest. In doing so, STARS ensures that the actual identity of a RSA which complies with the reputation management policies would never be disclosed, whereas a malicious RSA would be eventually isolated from the reputation system, and its identity will be revealed when it attempts to re-join.

What follows is the specific description of the protocol. Initially, let $(G_1, +)$ and $(G_2, \cdot)$ denote cyclic groups of prime order q, $P \in G_1$ a generator of G_1 and

Table 1. Reputation system entities and their parameters

	Private key	Public Key	Additional Parameter
PKG	s_{PKG}	Q_{PKG}	
RM	$s_{RM}, S_{RM} = s_{PKG} \cdot Q_{RM_2}$	$id_{RM}, Q_{RM_1} = s_{RM} \cdot P$	$Q_{RM_2} = H(id_{RM}, Q_{RM_1})$
QP	$s_{QP}, S_{QP} = s_{PKG} \cdot Q_{QP_2}$	$id_{QP}, Q_{QP_1} = s_{QP} \cdot P$	$Q_{QP_2} = H(id_{QP}, Q_{QP_1})$
Client	$s_{Client}, S_{Client} = s_{PKG} \cdot Q_{Client_2}$	$id_{Client}, Q_{Client_1} = s_{Client} \cdot P$	$Q_{Client_2} = H(id_{Client}, Q_{Client_1})$

let $e : G_1 \times G_1 \rightarrow G_2$ be a pairing which satisfies the properties of Bilinear and Non-degenerate. We also define the hash functions $H : \{0,1\}^* \times G_1 \rightarrow G_1$ and $h : \{0,1\}^* \times G_1 \rightarrow \mathbb{Z}_q^*$.

Setup. The PKG picks a random integer $s_{PKG} \in \mathbb{Z}_q^*$, computes $Q_{PKG} = s_{PKG}P$ and publishes Q_{PKG} while s_{PKG} is kept secret. The signer, which can be any entity of reputation system regardless of its particular role, also selects a random integer $s_1 \in \mathbb{Z}_q^*$ as its partially secret key and computes $Q_1 = s_1 P$ as its partially public key.

This algorithm is performed by the PKG when a signer requests the secret key corresponding to its identity. Suppose the signer's identity is given by the string id, which is the other partially public key. The other partially secret key of the identity is then given by $S_2 = s_{PKG}Q_2$ where $Q_2 = H(id, Q_1)$, which is computed by the PKG and given to the signer. For a signer, $\langle id, Q_1 \rangle$ is his public key and $\langle s_1, S_2 \rangle$ is his private key. The extraction step is typically done once for every identity and uses the same setup data for many different identities. Their corresponding keys are shown in Table 1, where Q_1 are specified as Q_{RM_1}, and Q_{QP_1} respectively.

Register. Client registers to RM in the following steps.

Step 1 (Client→RM): Send $m = righ \parallel id_{Client}$ to RM, where $right$ includes access right and time limitation of client id_{Client}

Step 2 (RM→Client):

- Verify if the $right$ complies with id_{Client}
- Choose random numbers $k, c, d \in \mathbb{Z}_q^*$
- Compute $a = e(Q_{RM_2}, Q_{PKG})^k$
  $B = c \cdot P + d \cdot Z$, $Z = H(right, id_{Client} \cdot P)$
- Send a, B to client

Step 3 (Client→RM):

- Choose random numbers $T_1 \in G_1, t_2, t_3, t_4 \in \mathbb{Z}_q^*$
- Compute $Z = H(right, id_{Client} \cdot P)$
  $\alpha = a \cdot e(T_1, P) \cdot e(Q_{RM_2}, Q_{RM_1})^{t_2}$
  $\beta = B + t_3 \cdot P + t_4 \cdot Z$
  $\gamma = h(\alpha \parallel \beta, Z)$
  $\tau = \gamma - t_2 - t_4 \bmod q$
- Send τ to RM

Step 4 (RM→Client):

- Compute $\nu = \tau - d$
  $U = k \cdot S_{RM} - s_{RM} \cdot \nu \cdot Q_{RM_2}$
- Send U, ν, c, d to Client

Step 5 (Client→RM):

- Compute $\Omega = U + T_1$

$\omega = \nu + t_2 \bmod q$
$\Delta = (c + t_3) \cdot P$
$\delta = d + t_4 \bmod q$

- Verify $\omega + \delta \stackrel{?}{=} h(e(\Omega, P) \cdot e(Q_{RM_2}, Q_{RM_1})^{\omega} \parallel (\Delta + \delta \cdot Z), Z)$ mode q and $Q_{RM_2} \stackrel{?}{=} H(id_{RM}, Q_{RM_1})$
  If it is true, continue; otherwise, abort.

Login. Client logs in to QP to query reputation values.
Step 1 (Client→QP): Send $right, id_{Client} \cdot P, \Omega, \omega, \Delta, \delta$ to QP.
Step 2 (QP→ Client):

- Check the time limitation in $right$
- Compute $Z = H(right, id_{Client} \cdot P)$
- Verify $\omega + \delta \stackrel{?}{=} h(e(\Omega, P) \cdot e(Q_{RM_2}, Q_{RM_1})^{\omega} \parallel (\Delta + \delta \cdot Z), Z)$ mode q and $Q_{RM_2} \stackrel{?}{=} H(id_{RM}, Q_{RM_1})$
  If it is yes, continue; otherwise, abort.
- Send x to Client, x is a random number generated by QP.

Step 3 (Client→QP):

- Set $R_1 = id_{Client} \cdot P$
  $\nu_1 = h(x, R_1)$
  $u_1 = id_{Client} - \nu_1(c + t_3)$
- Send ν_1, u_1 to QP

Step 4 (QP→Client):

- Verify $\nu_1 \stackrel{?}{=} h(x, u_1 \cdot P + \nu_1 \cdot \Delta)$
  If equals, the client is allowed to access the reputation system via QP; otherwise, abort.

Trace. If an entity registers RM more than once, it's real identity can be revealed in the following way.
Step 1 (Client→RM): Send $right, id_{Client} \cdot P, \Omega, \omega, \Delta, \delta$ to RM.
Step 2 (RM→Client):

- Check the time limitation in $right$.
- Compute $Z = H(right, id_{Client} \cdot P)$
- Verify $\omega + \delta \stackrel{?}{=} h(e(\Omega, P) \cdot e(Q_{RM_2}, Q_{RM_1})^{\omega} \parallel (\Delta + \delta \cdot Z), Z)$ mode q and $Q_{RM_2} \stackrel{?}{=} H(id_{RM}, Q_{RM_1})$
  If it is yes, continue; otherwise, abort.
- Send y to Client, y is a random number generated by RM

Step 3 (Client→RM):

- Set $R_2 = id_{Client} \cdot P$
  $\nu_2 = h(y, R_2)$
  $u_2 = id_{Client} - \nu_2(c + t_3)$

- Send ν_2, u_2 to RM

Step 4 (RM→Client):

- Verify $\nu_2 \stackrel{?}{=} h(y, u_2 \cdot P + \nu_2 \cdot \Delta)$
  If it is yes, the client is authorized to access system; otherwise, abort.

Step 5 Identify misbehavior;
Step 6 Reveal id_{Client}: $u_1 \cdot P + \nu \cdot (c + t_3) \cdot P = R_1 = R_2 = u_2 \cdot P + \nu_2 (c + t_3) \cdot P$, such that,

$$u_1 + \nu_1(c + t_3) = u_2 + \nu_2 \cdot (c + t_3) \bmod q \Rightarrow \begin{cases} c + t_3 = \frac{u_2 - u_1}{\nu_1 - \nu_2} & \bmod q \\ id_{user} = \frac{\nu_1 \cdot u_1 - \nu_2 \cdot u_2}{\nu_1 - \nu_2} & \bmod q \end{cases}$$

In doing so, the identity of misbehaving node can be identified.

5 Performance Validation

This section discusses implementation issues and reports performance validation through case study, comparative analysis, and simulation.

5.1 Security Analysis

Anonymity. Considering the register process, we can prove that anonymity for honest users is unconditional. We follow the formal security model in [5]. Given a valid signature $(\Omega, \omega, \Delta, \delta, m)$ and any view (U, ν, c, d, τ), the following equations then hold for $T_1 \in G_1$, $t_2, t_3, t_4 \in \mathbb{Z}_q^*$:

$$\Omega = U + T_1 \tag{1}$$
$$\omega = \nu + t_2 \bmod q \tag{2}$$
$$\Delta = (c + t_3) \cdot P \tag{3}$$
$$\delta = d + t_4 \bmod q \tag{4}$$
$$\tau = \nu + d \tag{5}$$

$$\tau = h(a \cdot e(T_1, P) \cdot e(Q_{RM_2}, Q_{RM_1})^{t_2} \| B + t_3 \cdot P + t_4 \cdot Z, Z) - t_2 - t_4 \bmod q \tag{6}$$

Thus, the following values can be calculated:

$$T_1 = \Omega - U \tag{7}$$
$$t_2 = \omega - \nu \bmod q \tag{8}$$
$$t_3 = \log_p \Delta - c \bmod q \tag{9}$$
$$t_4 = \delta - d \bmod q \tag{10}$$

Next, we prove the equations (5) (6) hold:

$$\omega + \delta = h(e(\Omega, P) \cdot e(Q_{RM_2}, Q_{RM_1})^{\omega} \| (\Delta + \delta \cdot Z), Z) \bmod q$$

$$\Leftrightarrow \nu + d + t_2 + t_4 = h(e(\Omega, P) \cdot e(Q_{RM_2}, Q_{RM_1})^{\omega} \| (\Delta + \delta \cdot Z), Z) \bmod q$$

$$\Leftrightarrow \nu + d + t_2 + t_4 = h(a \cdot e(T_1, P) \cdot e(Q_{RM_2}, Q_{RM_1})^{t_2} \| B + t_3 \cdot P + t_4 \cdot Z, Z) \bmod q$$

$$\Leftrightarrow \nu + d = h(a \cdot e(T_1, P) \cdot e(Q_{RM_2}, Q_{RM_1})^{t_2} \| B + t_3 \cdot P + t_4 \cdot Z, Z) - t_2 - t_4 \bmod q$$

Therefore, the blind factors always exist which lead to same relation defined in register protocol. Even a traitor with infinitely computation power can win the game defined in [5] with a negligible advantage. Thus, neither RM nor QP can associate a legitimate client's login process to its real identity.

Traceability. If a client has any misbehavior defined above, the anonymity will not be preserved any more. In order to escalate the privilege, a traitor may try to implement the login protocol more than once, thereby using the register information twice or more. Through trace protocol, the traitor's real identity can be disclosed.

Unforgeability. The proof is similar to the proof of Theorem 5.3 in [17] by deducing the security of the new scheme from the hardness of the CDHP in random oracle model, the new scheme is proved to satisfy the unforgeability.

Henceforth, we can claim that STARS satisfies the security requirements on anonymity, traceability, and unforgeability. Also, in order to examine the operational overhead of STARS, we simply count its major computations. Generally, the pairing operation is several times more complex than the scalar multiplication in G_1, thus the number of the pairing operation is a key performance metric. In particular, the following computations are considered: $P-$the number of pairing operation, $E-$ the number of exponentiation in G_2, $M-$ the number of multiplication in G_1, and $H/h-$the number of hash operation. The total computational overhead is summarized in Table 2.

Table 2. Operational Overhead of STARS

	RM/QP				Client			
Computation	P	E	M	H/h	P	E	M	H/h
Register	1	1	5	1	2	1	3	2
Login	2	1	3	4	0	0	0	1
Trace	2	1	3	4	0	0	0	1

5.2 Implementation and Qualitative Analysis

Regardless of the specific operational characteristics of underlying reputation systems (RS), a general implementation procedure of STARS can be described by Algorithm 1.

Initialization RS, PKG
Select RMs, QPs; //They can be updated by RS later;
Setup;
foreach *network node which intends to join in RS* **do**
  Register to RM_i;
  Login to QP_i;
end
Run RS;
if *the client cl isolated by* RS *attempts to re-join* **then**
  Trace cl;
end

Algorithm 1. Integrating STARS with RS

L: Low M: Medium H: High	Computational overhead	Sensitivity	Scalability	Adaptability	Accuracy	Security and privacy	Robustness	Application scenario
eBay [8]	L →	L →	H →	H →	L →	L ↗	H →	E-commerce
SuperTrust [5]	H ↘	M →	M →	M →	M →	H ↗	H →	P2P
CONFIDENT [3]	H ↘	M →	H ↘	H ↘	L →	L ↗	M ↘	MANET
TrustMe [23]	M →	M →	M →	M →	M →	M ↗	M ↘	P2P
EigenTrust [24]	M ↘	H →	M ↘	H →	M →	M ↗	H →	P2P

Performance Enhanced ↗ Performance Deteriorated ↘ Performance Maintained →

Fig. 3. Performance comparison without (with) the introduction of STARS

To evaluate the feasibility and impact of integrating STARS with RS, we carefully examined a number of representative RSs (which are essentially ARS) that work in different scenarios, including eBay [9], CONFIDENT [4], SuperTrust [6], TrustMe [25], and EigenTrust [27], and qualitatively analyze their performance in terms of the operational metrics proposed in Sec. 3.1 for identifying whether (why, and how) they are improved or undermined due to the introduction of STARS. The results are briefly summarized in Fig. 3, based on a comparative study between RS candidates. Specifically, we have three major observations:

– *Security and Privacy* of all the RSs can be enhanced thanks to STARS, although extra computational overhead are possibly incurred. The enhanced RSs are particularly resilient to sybil attack, which intends to create a large number of fake pseudonyms. This is because one client can only have one pseudonym while multiple (more than once) registration would lead its actual identity to be disclosed.

– Since STARS is transparent to the underlying RS, it does not incur negative impact on the *sensitivity* and *accuracy* of RSs (in that the identity of the anomalous node which is isolated from RS will be disclosed and obliged to re-register), but the *scalability*, *adaptability*, and *robustness* may vary in different operational environments: CONFIDENT is originally designed for MANET which is fully distributed, its scalability, adaptability and robustness would be slightly undermined due to the introduction of STARS which requires pre-trusted nodes and hence introduces the singular point of failure, as the first step of Algorithm 1 implies. However, the highly centralized RS like that of eBay and those RSs tailored for P2P networks would not suffer performance deterioration on those metrics.
– The comparative analysis generally suggests that STARS has potential to be applied in heterogeneous autonomic networks with pre-trusted RM, and the trade-offs between operational metrics can be carefully tuned for attaining the optimum. In particular, the reputation systems which have already been developed for P2P networks like EigenTrust [27] and SuperTrust [6] can be enriched with the functionality of *traitor tracing* and further deployed in autonomic networks. One prerequisite, however, is that their computational overhead should be kept to be minimal comparing with the reputation management without STARS.

5.3 Simulation-Based Quantitative Evaluations

While the majority of operational metrics can be qualitatively analyzed, we are interested in examining the extent and root causes of negative consequence caused by STARS on *computational overhead* and *scalability* by conducting simulations and quantitative evaluations. To keep simplicity without loss of generality, we simulated STARS with eBay's reputation system [9] and CONFIDENT [4] using NS2 [23].

Evaluation Metrics. We introduce ρ_{co} for quantifying *computational overhead*. As the computational complexity of STARS has been analyzed in Table 2, we are particularly concerned with the messaging cost incurred by STARS. Formally, we define it as $\rho_{co} = \#Msg'/\#Msg$, where $\#Msg'$ is the number of messages associated with STARS's operations, and $\#Msg$ is the total number of messages caused by RS. We also examine the relationships between $\#Msg$ and $\#RM$ (the number of RMs) for measuring *scalability*.

Simulation Settings. We randomly deploy an MANET in a space of $1000\times1000\ m^2$, running DSR routing protocol. The application layer uses CBR for generating data packets ($6/sec.$). 50 source-destination connections are randomly and periodically (10 seconds) generated among the nodes. Also, Random Waypoint Mobility Model is used to simulate the node mobility, and the node speed is randomly generated between 0 and 20 m/s.

Findings. We observed that STARS did not cause much extra cost except the *register* and *query* phases, while *setup* can be done together with the deployment

Table 3. Simulation settings

Network	Area	1000m × 1000m
	Topology	random
	Placement of nodes	uniform
	Routing protocol	DSR
RS	with (without) STARS	CONFIDENT [4], eBay [9]
Node	# of nodes	100
	# of malicious nodes	variable
	# of RM	variable
	Node's initial reputation	0.6
Anomalies	DoS	Packet dropping
	Selfish	Refusing to forward packets
Simulation	# of simulation epochs	20
	running time	5000s
	Length of observation window (l_{ow})	20s

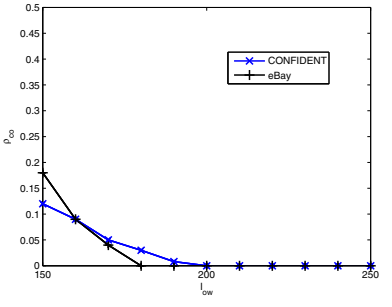


Fig. 4. The ratio between the messaging cost of STARS with that of RSs

of such RSs which require bootstrap process like eBay and TrustMe [25]. In particular, Fig. 4. plots the ρ_{co} of eBay and CONFIDENT, implying that centralized RS has an overhead burst at the initial running of STARS (plain RSs ran in the first 150 observation windows) at the 150-th observation window, but it converged to 0 faster than that of CONFIDENT.

On average, STARS with CONFIDENT had higher ρ_{co} than that of eBay. This is not surprising considering the fact that a portion of nodes left and re-joined CONFIDENT frequently. It is thus conservative to claim that the introduction of STARS would lead to higher computational overhead in distributed reputation systems. It is also observed that algorithm *Trace* did not incur extra cost for either eBay or CONFIDENT. Based on these findings, it is reasonable to assume that the extra overhead caused by STARS can be minimized by setting appropriate query time, because the number of messages incurred by STARS is neglectable compared to that of messages for querying trust ratings, while the two types of messages can be simply merged together.

We also examined the implicit relationship between ρ_{co} and the number of RMs. In terms of this metric, STARS had higher impact on CONFIDENT than eBay on *scalability*. Indeed, the previous simulation result has implied this fact: for centralized RSs, increasing the number of RM to an appropriate threshold

may reduce ρ_{co} as a whole, but it is not always true in the distributed operational environment like P2P networks. Furthermore, we found that *Trace* process may trigger false positives when a legitimate entity re-joins the system with the same identity, and it varied with the percentage of anomalous entities in reputation systems. In general, CONFIDENT has higher false positive than eBay.

6 Concluding Remarks

In this paper, we gave a comprehensive examination on the typical reputation systems by decomposing them into four functional components and manifesting their respective vulnerabilities. We then showed that it is feasible for a sophisticated insider to subvert a reputation system by exploring the implicit relationship between node identity and its reputation values. To prevent such attack, we developed STARS to preserve both anonymity and traceability in reputation systems, providing an interface to balance the operational metrics of concern in different operational environments. Although some implementation issues have been addressed, and some comparative studies have been conducted, it is still desirable to enrich quantitative evaluations via extensive simulations for validating the performance of STARS on detecting inside attackers (traitors) in a specific network scenario. We keep those as future work. Also, we intend to further quantify the implicit association between anonymity and traceability in more fine-grained ways.

References

1. Androulaki, E., Choi, S., Bellovin, S.M., Malkin, T.: Reputation Systems for Anonymous Networks. In: Borisov, N., Goldberg, I. (eds.) PETS 2008. LNCS, vol. 5134, pp. 202–218. Springer, Heidelberg (2008)
2. Adams, W.J., Hadjichristofi, G.C., Davis, N.J.: Calculating a node's reputation in a mobile ad hoc network. In: Proc. of Int'l Performance Computing and Communications Conference, AZ (April 2005)
3. Bethencourt, J., Shi, E., Song, D.: Signatures of Reputation: Towards Trust Without Identity. In: Sion, R. (ed.) FC 2010. LNCS, vol. 6052, pp. 400–407. Springer, Heidelberg (2010)
4. Buchegger, S., Le Boudec, J.-Y.: Performance analysis of the CONFIDANT protocol. In: Proc. of ACM MobiHoc, Lausanne, Switzerland, pp. 226–236 (2002)
5. Chow, S.M., Hui, L.C.K., Yiu, S.M., Chow, K.P.: Two improved partially blind signature schemes from bilinear pairings. In: Boyd, C., González Nieto, J.M. (eds.) ACISP 2005. LNCS, vol. 3574, pp. 316–328. Springer, Heidelberg (2005)
6. Dimitriou, T., Karame, G., Christou, I.: SuperTrust: a secure and efficient framework for handling trust in super-peer networks. In: Proc. of ACM PODC, pp. 374–375 (2007)
7. Dingledine, R.: Accountability Measures for Peer-to-Peer Systems. In: Peer-to-Peer: Harnessing the Power of Disruptive Technologies. O'Reilly Publishers, Sebastopol (2000)
8. Dingledine, R., Mathewson, N., Syverson, P.: Tor: The Second-Generation Onion Router. In: Proc. of 13th USENIX Security Symposium, pp. 303–320 (2004)

9. http://www.ebay.com
10. Freedman, M.J., Morris, R.: Tarzan: A peer-to-peer anonymizing network layer. In: Proc. of CCS 2002, Washington, DC, USA, pp. 193–206 (2002)
11. He, Q., Wu, D., Khosla, P.: SORI: A secure and objective reputation-based incentive scheme for ad hoc networks. In: Proc. of Wireless Communications and Networking Conference, pp. 825–830 (2004)
12. Huebscher, M.C., Mccann, J.A.: A survey of autonomic computing-degrees, models, and applications. ACM Computing Surveys 40(3) (August 2008)
13. Hoffman, K., Zage, D., Nita-Rotaru, C.: A Survey of Attack and Defense Techniques for Reputation Systems. ACM Computing Surveys (2008)
14. Josang, 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. Kerschbaum, F.: A verificable, Centralized, Coercion-Free Reputation System. In: Proc. of Workshop on Privacy in the Electronic Society (WPES), USA (2009)
16. Liu, Y., Yang, Y.R.: Reputation propagation and agreement in mobile ad-hoc networks. In: Proc. of IEEE Wireless Communications and Networking (WCNC 2003), New Orleans, USA (2003)
17. Liu, J., Sun, R., Kou, W., Wang, X.: Efficient ID-based Signature Without Trusted PKG, http://eprint.iacr.org/2007/135.pdf
18. Marmol, F.G., Perez, G.M.: Security threats scenarios in trust and reputation models for distributed systems. Computers & Security 28(7), 605–614 (2009)
19. Marti, S., Garcia-Molina, H.: Identity crisis: anonymity vs reputation in P2P systems. In: Proc. of the Third International Conference on Peer-to-Peer Computing (P2P 2003), pp. 134–141 (September 2003)
20. Miranda, H., Rodrigues, L.: A framework to provide anonymity in reputation systems. In: Proc. of MOBIQUITOUS 2006 (2006)
21. Muller, W., Plotz, H., Redlich, J.-P., Shiraki, T.: Sybil proof anonymous reputation management. In: Proc. of ACM SecureComm 2008 (September 2008)
22. Mundinger, J., Le Boudec, J.-Y.: Analysis of a reputation system for mobile ad-hoc networks with liars. In: Proc. of The 3rd International Symposium on Modeling and Optimization, Trento, Italy (April 2005)
23. Nework Simulator, http://www.isi.edu/nsnam/ns/
24. Song, S., Hwang, K., et al.: Trusted P2P transactions with fuzzy reputation aggregation. IEEE Internet Computing 9(6), 24–34 (2005)
25. Singh, A., Liu, L.: TrustMe: Anonymous Managment of Trust Relationships in Decentralized P2P systems. In: Proc. of 3rd International IEEE Conference on Peer-to-Peer Computing (2003)
26. Steingrecher, S.: Design options for privacy-respecting reputation systems within centralised internet communities. In: Proc. of Int. Information Security Conference, SEC (2006)
27. Kamvar, S.D., Schlosser, M.T., Molina, H.G.: The EigenTrust algorithm for reputation management in P2P networks. In: Proc. of the 12th International Conference on World Wide Web, pp. 640–651 (2003)
28. Yu, B., Singh, M.P.: An evidential model of distributed reputation management. In: Proc. of ACM AAMAS (2002)
29. Zouridaki, C., Mark, B.L., Hejmo, M., Thomas, R.K.: Hermes: A auantitative trust establishment framework for reliable data packet delivery in MANETs. Journal of Computer Security 15(1), 3–38 (2007)
30. Zhang, Y., Liu, W., Lou, W.: Anonymous communications in mobile ad hoc networks. In: Proc. of IEEE INFOCOM, Miami, USA (2005)

31. Zhang, Z., Nait-Abdesselam, F., Ho, P.-H., Lin, X.: RADAR: a ReputAtion-based scheme for Detecting Anomalous nodes in wiReless mesh networks. In: Proc. of IEEE Wireless Communications and Networking Conference (WCNC 2008), Las Vegas, USA (2008)
32. Zhang, Z., Kadobayashi, Y., Nait-Abdesselam, F.: Towards an Evaluation Framework for Reputation Systems in Autonomic Networks. In: Proc. of ChinaCom 2009, Xi'An, China, August 26-28 (2009)
33. Zhou, L., Haas, Z.: Securing ad hoc networks. IEEE Network 13(6), 24–30 (1999)
34. Zhong, S., Chen, J., Yang, R.: Sprite: a simple, cheat-proff, credit-based system for mobile ad-hoc networks. In: Proc. of IEEE INFOCOM, San Francisco, USA (2003)

DualTrust: A Distributed Trust Model for Swarm-Based Autonomic Computing Systems

Wendy Maiden[1,2], Ioanna Dionysiou[2,3], Deborah Frincke[1,2], Glenn Fink[1], and David E. Bakken[2]

[1] Pacific Northwest National Laboratory, Richland, WA
{wendy.maiden,deborah.frincke,glenn.fink}@pnl.gov
[2] School of Electrical Engineering and Computer Science
Washington State University, Pullman, WA, USA
bakken@eecs.wsu.edu
[3] Department of Computer Science
University of Nicosia, Nicosia, Cyprus
dionysiou.i@unic.ac.cy

Abstract. For autonomic computing systems that utilize mobile agents and ant colony algorithms for their sensor layer, trust management is important for the acceptance of the mobile agent sensors and to protect the system from malicious behavior by insiders and entities that have penetrated network defenses. This paper examines the trust relationships, evidence, and decisions in a representative system and finds that by monitoring the trustworthiness of the autonomic managers rather than the swarming sensors, the trust management problem becomes much more scalable and still serves to protect the swarm. We propose the DualTrust conceptual trust model. By addressing the autonomic manager's bi-directional primary relationships in the ACS architecture, DualTrust is able to monitor the trustworthiness of the autonomic managers, protect the sensor swarm in a scalable manner, and provide global trust awareness for the orchestrating autonomic manager.

Keywords: Trust management, autonomic computing systems, reputation, mobile agent, DualTrust.

1 Introduction

Complementing traditional security techniques that emphasize authentication and the prevention of security failures, trust management uses reputation to detect potentially malicious quality of service (QoS) issues (e.g., resource starvation) [1] and uses authorization credentials for powerful, yet lightweight, authorization and control. This detection capability is critical when the agents are part of a security system that must withstand the attacks of those seeking to thwart the system's traditional security measures, including those who have penetrated external defenses and those who are insiders. Actively managing trust rather than depending on blind trust also serves to improve acceptance of mobile agent-based systems.

J. Garcia-Alfaro et al. (Eds.): DPM 2010 and SETOP 2010, LNCS 6514, pp. 188–202, 2011.

For autonomic computing systems (ACSs) that utilize mobile agent sensors patterned after an ant colony swarm [2], [3], three key aspects of the system architecture must be considered in designing the trust model – the characteristics of the mobile agent swarm, the peer interactions of the autonomic managers, and the policy-oriented oversight provided by the orchestrating autonomic manager. This paper characterizes unique trust requirements of swarm-based autonomic computing systems, discusses the applicability of existing research efforts, describes trust relationships, evidence, evaluation, and decisions in a representative swarm-based ACS, and introduces DualTrust, a conceptual trust management model for swarm-based ACSs that meets the specified requirements.

DualTrust is a trust model that reflects the autonomic manager's bi-directional primary relationships in the ACS architecture. By monitoring the trustworthiness of autonomic managers as they interact with the sensors, DualTrust benefits the swarm and the orchestrating autonomic manager as well.

The remainder of this paper is organized as follows: Section 2 provides an overview of ACSs and describes a representative example of an ACS that uses a mobile agent sensor swarm. Section 3 analyzes trust considerations in an ACS, with comments on related work. Section 4 introduces DualTrust. Section 5 discusses conclusions and future research directions.

2 Autonomic Computing Systems

This section provides an overview of ACSs in general and an overview of an example swarm-based ACS.

2.1 Overview of Autonomic Computing Systems

ACSs provide automated, flexible, context-aware application of human-derived policy to the overall maintenance of computer systems [4]. The general architecture of an ACS is a set of cooperating autonomic elements [4], [5]. Each autonomic element has two parts – the managed element, such as a computer system or legacy software, and the autonomic manager. The autonomic manager provides the autonomic (i.e., self-regulating) behavior of the element, utilizing sensors to probe the state of the managed element to determine if it is still functioning according to policy and utilizing effectors to configure and maintain it as needed. Some ACSs have an orchestrating autonomic manager that resides hierarchically above the autonomic elements. The orchestrating autonomic manager translates and communicates policy to the autonomic managers and may collaborate with orchestrating autonomic managers from other ACSs.

2.2 CID as a Swarm-Based ACS

The Cooperative Infrastructure Defense (CID) framework is being developed by the Pacific Northwest National Laboratory as an alternate model for securing complex cyber infrastructures that have multiple owners with potentially competing policy requirements [1], [6]. An infrastructure is defined to be a

multi-organizational entity whose elements share a unified purpose, but who do not necessarily have the same policies, hardware, or ownership. A prime example is an electrical power grid where companies in the grid may share management, maintenance, and housing of computer, network, and Supervisory Control and Data Acquisition (SCADA) resources for the unified purpose of providing reliable power to a large geographic area. CID is an autonomic, cyber-defense framework designed to allow multiple organizations to cooperate in cyber defense of an infrastructure, while respecting proprietary interests and requiring minimal human intervention or direction.

CID enables many characteristics [4] of an ACS, such as self-protection and self-healing. However, CID concentrates on adaptive security across an infrastructure as opposed to general maintenance of an individual machine or enclave. The CID framework uses a hierarchy of rational agents to monitor hosts within and across enclaves (i.e., security domains).

At the top of the CID hierarchy (see Fig. 1) is a human Supervisor who provides the policy and is ultimately responsible for the actions of the CID agents. At the next level down, the Sergeant agent is responsible for maintaining overall enclave security. The Sergeant is the orchestrating autonomic manager. It dialogues with the Supervisor to create executable policy statements[1] to pass to the autonomic managers. The Sergeant also controls the Geography, which is the set of authorized autonomic elements in the ACS.

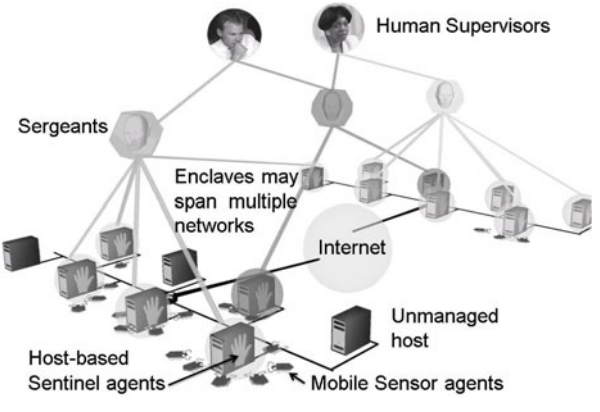


Fig. 1. CID hierarchy. Entities from top to bottom are: The human *Supervisors*, *Sergeants*, host-based *Sentinels*, and mobile *Sensors*

Each autonomic element in CID consists of a host computer as the managed element and a Sentinel as its autonomic manager [5]. The Sentinel provides

[1] The policy language is not specified in this paper because it is determined by the ACS and is not determined or constrained by DualTrust. DualTrust provides trust data which the ACS uses to evaluate trust-based policy statements. Section 4.6 provides examples of trust-based policy.

the effectors while the sensors are provided by swarming Sensor[2] agents that adaptively sense conditions across the entire enclave using a modified ant colony algorithm [2], [3] to sense and affect system stability. Although individual Sensor communications are limited to local, stigmergic messaging, the overall effect creates useful emergent behaviors that characterize the swarm as a whole [7].

Additional information on CID can be found in [6], and an analysis of trust in CID can be found in [1] and [8].

3 Trust Considerations in an ACS

This section provides an overview of the trust management issues and relationships for each of the three types of entities in an ACS – the sensor swarm, the autonomic managers, and the orchestrating autonomic manager, and provides comments on related research.

3.1 Protecting the Mobile Agent Sensor Swarm and Their Hosts

Hosts can be protected from malicious sensor agents in large part through well-documented, standard security mechanisms [1], [9]. Protecting the mobile agent sensor swarm from corrupted hosts is a more challenging problem.

Unlike the typical mobile agent trust research scenario [10], [11], [12], swarming CID Sensors have an open-ended mission rather than a fixed itinerary; they communicate only indirectly through pheromone; and they must not avoid going to compromised hosts so long as the host's Sentinel (autonomic manager) is still trusted. They also are lightweight, numerous, and relatively ephemeral. None of the surveyed mobile agent trust frameworks would accommodate these characteristics. Although it must be modified [1] for use with the Sensor swarm, the work of Lin et al [11], [13], [14] provides the applicable concept of checking the reputation of hosts on the prospective itinerary before sending a mobile agent and checking the reputation of prior hosts before receiving a mobile agent.

3.2 Monitoring and Protecting the Autonomic Managers

Unlike peer-to-peer systems, autonomic managers of an ACS interact vertically across multiple levels (with the swarming sensors and with the orchestrating autonomic manager), not just horizontally with their autonomic manager peers. Therefore the trust model must take into consideration the need for these other entities to access and update trust data. Xiong and Liu's P2P-inspired trust storage and distribution mechanism [15] and Stakhanova, et al.'s use of anomaly detection [16] to externally collect evidence to improve trust calculations are useful concepts that can be adapted for use with the autonomic managers.

[2] The word *sensor* refers to the sensor function in an ACS. In contrast, *Sensor* refers to the proper title of the entity in CID that implements the sensor function.

3.3 Providing Global Trust Awareness to the Orchestrating Autonomic Manager

In a hierarchical distributed system, the higher-level nodes provide a natural location at which to place a trust evidence collection and evaluation role. Pushing trust evidence upward provides global trust awareness at the highest tier, e.g., the orchestrating autonomic manager in an ACS. A hierarchical trust model minimizes overall communication if appropriate filtering or pre-processing is done at lower levels.

4 DualTrust: A Distributed Trust Model for Managing the Trust of Autonomic Managers

This section introduces the DualTrust conceptual trust model. It discusses the trust foundations for DualTrust and the architectural design requirements that the trust model was designed to meet. It then introduces the two halves of DualTrust – the horizontal and vertical evidence storage and distribution frameworks. The trust evaluation calculation is introduced next, followed by a scenario to demonstrate how the trustworthiness of key trust relationships is improved by using DualTrust.

4.1 DualTrust Foundations

The DualTrust model uses both credentials and reputation as the foundation for determining trustworthiness. This section discusses the authentication, authorization credentials, and reputation evidence that form the foundation for trust in DualTrust.

Authentication. Supervisors, Sergeants, and Sentinels are each assigned a public/private key pair for authentication. Once they have been authenticated, Supervisors and Sergeants are treated as trusted entities.

Each Sentinel has the public key of the Sergeant and the other Sentinels preloaded. Subsequent public key changes (such as the addition of a public key for a new Sentinel) are passed down to the Sentinels by the Sergeant.

Because the Sensors are numerous and ephemeral, the associated key management functions (e.g., key creation, constant distribution of public keys for new Sensors, and constant issuance of updated revocation lists) would add significant overhead without a corresponding significant increase in trust. Therefore, Sensors are not assigned an identifying key pair and are not otherwise uniquely identified. Instead, each Sensor carries a signed authorization credential from the Sentinel that created it. A Sensor can be trusted if the Sentinel that created the Sensor is trusted and if the Sensor's code has not changed since it was created.

Authorization Credentials. Two forms of authorization are used in Dual Trust. First, Sentinels carry an authorization credential (such as a SPKI/SDSI credential [17]) containing the ID of the Sergeant and digitally signed with the

Sergeant's private key to enable verification of source and content integrity. Sensors carry a similar authorization credential containing the ID of the creating Sentinel, and digitally signed by the creating Sentinel.

The second form of authorization credential is the Geography managed by the Sergeant. The Geography (i.e., the set of hosts that the Sensor agents are allowed to visit) functions as an inverse credential revocation list (i.e., a whitelist). If an agent has a Sentinel role credential, it only proves that the agent was granted the credential at some point in the past. However, the existence of the Sentinel in the current Geography (as received from and signed by the Sergeant) confirms its continued authorization. When a Sergeant removes an offending Sentinel from the Geography and publishes the revised Geography to the remaining Sentinels, the Sentinel's authorization is revoked. Sentinels check the Geography as part of the authorization process and will terminate Sensors sent or created by the banned Sentinel, thus protecting the hosts and their Sentinels, and will no longer send Sensors to the banned Sentinel, thus protecting the Sensors.

Reputation Evidence. There are two types of reputation evidence used in DualTrust – complaints and quality of service (QoS) observations. Quality of service (QoS) observations consist of the following elements:

- `By_SID` is the ID of the Sentinel that observed and reported the evidence,
- `About_SID` is the ID of the Sentinel about which the trust evidence was reported,
- `DT` is the date and time of the trust evidence,
- `Context` is the trust evidence category (Sensor_Integrity, Sensor_Resourcing, Sensor_Policing, etc),
- `Pass/Fail` is the feedback, where Pass = 1 and Fail = 0, and
- `Signed_Hash` is a cryptographic hash on the first five elements that the originating Sentinel digitally signs with its private key.

The complaint data structure is similar to that of QoS observations but does not have the `Pass/Fail` parameter since a complaint is only used to report a failure to meet critical expectations.

4.2 Architectural Design Requirements

The following requirements constrain the choice of architecture for a swarm-based ACS:

- It must focus on the trustworthiness of the persistent autonomic entities rather than the more abundant and ephemeral sensor entities [1].
- Both the autonomic managers and the orchestrating autonomic manager must be able to access each autonomic manager's trust data.
- Complete trust data must be readily available. Depending on a subset of trust data experienced by an autonomic manager and its neighbors is insufficient, because neighbors will only know about certain contexts and will be slow to learn about damage caused elsewhere.

– It must be lightweight and scalable.
– It must be fault tolerant. For example, it must maintain its stability and accuracy when systems are shut down or malicious trust data is provided.
– It must not encumber agent adaptation.

4.3 The Horizontal Dimension of DualTrust Evidence Storage and Distribution

Xiong and Liu's PeerTrust [15] provides the inspiration for the distributed trust model used between the Sentinels. In PeerTrust, trust evidence is routed for storage just as P2P files are routed for storage, and trust evidence requests are routed for fulfillment just as P2P file requests are routed. Each peer stores a full set of trust data for one or more other peers, and each peer's trust data is stored on one or more other peers to allow a voting process to be used to detect evidence tampering.

Because they are known entities within a single enclave, Sentinels have no need of a P2P-style routing mechanism. Instead, they directly contact the storing Sentinel to gather or write reputation data. The Sergeant, as part of its policy responsibilities, prescribes the trust storage locations for each node. This design has the benefit of making complete reputation data readily available to the Sentinels and also to the Sergeant.

Fig. 2 shows a reputation evidence collection scenario and the Geography that will be used for illustration throughout this section. The details of Sentinel X's internals pertaining to evidence collection are also shown. The grid represents the Geography, where each square in the Geography represents a Sentinel. In this scenario, Sentinel X is preparing to send a Sensor to neighboring Sentinel Y. Y's reputation evidence is stored on Sentinel Z. The Geography also includes Sentinels A, B, and C and other Sentinels that are not labeled in the figure. The following steps illustrate the evidence collection scenario.

Scenario. Sentinel X collects reputation evidence for Sentinel Y prior to passing a Sensor to Sentinel Y.

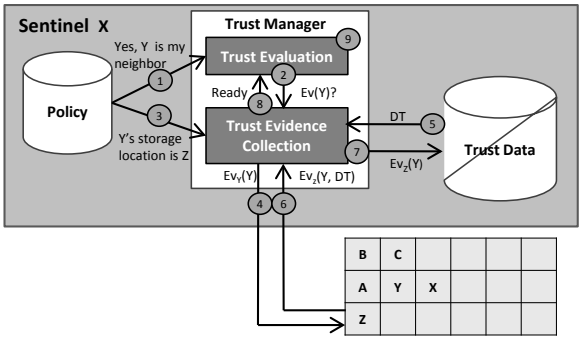


Fig. 2. Reputation evidence collection scenario

1. The Trust Evaluation module first checks its copy of the Geography, which is stored in its Policy database to ensure that Sentinel Y is both a member of the Geography and a neighbor of X in the Geography.
2. If it is, it asks the Trust Evidence Collection module to gather Y's reputation evidence.
3. The Trust Evidence Collection module checks the Geography, stored in the Policy database, to determine which Sentinel(s) (in this case, Z) store Y's evidence.
4. The Trust Evidence Collection module proceeds with any direct reputation observations it can make and writes the evidence it observed to Z.
5. The Trust Evidence Collection module checks its Trust Data store to determine if it contains any previously collected evidence for Y. It notes the latest timestamp, DT, of the evidence data.
6. The Trust Evidence Collection module requests evidence newer than date/time DT from Z.
7. The trust evidence for Y obtained from Z, $Ev_z(Y)$, is written to the local Trust Data store.
8. The Trust Evidence Collection module then notifies the Trust Evaluation module that the evidence data is available for evaluation.
9. The Trust Evaluation module calculates the reputation score.

To reduce the amount of trust evidence being transported over the network, Sentinels store evidence they previously retrieved, so they only need to retrieve the evidence gathered since the previous request for evidence about that node. The Sergeant uses policy statements to authorize Sentinels that already have a reputation score or trust evidence for a given peer, to skip the refresh request if the data is no older than a specified interval and the score for the peer is above a specified threshold that is higher than the authorization threshold. For example, if the authorization threshold is .90, then the Sergeant may permit Sentinels to use a cached reputation score if the score is $> .95$ and was calculated no more than 15 minutes ago.

To minimize network traffic further, the Sentinel can request just the reputation score (signed for non-repudiation and to prevent undetected, in-transit tampering) from the storing Sentinels. If all scores are above the threshold required for authorization, the average score can be used. Otherwise, evidence records are retrieved from each storing Sentinel, using the voting mechanism to determine the best data to use. If a storing Sentinel is found to have tampered with the data, reputation evidence with the context of Evidence_Tampering will be recorded for that Sentinel.

4.4 The Vertical Dimension of DualTrust Evidence Storage and Distribution

To avoid scalability issues, DualTrust uses a complaint-based model (see Fig. 3) that parallels the process where consumers register complaints about untrustworthy businesses with the Better Business Bureau [18] in the United States. Aberer and Despotovic [19] recommend a complaint-based model when it can

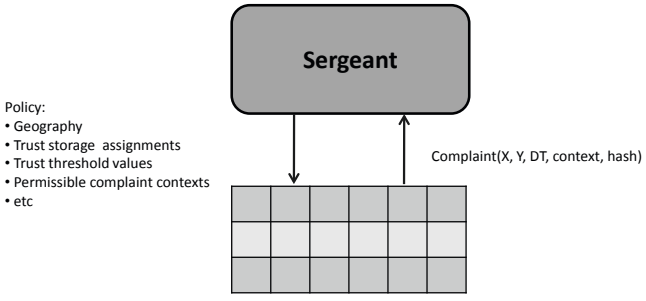


Fig. 3. The Sergeant's hierarchical relationship to the Sentinels

be expected that trust usually exists and malicious behavior is the exception, which can be expected to be the case in a managed, intra-domain environment. Because positive feedback is not recorded, the network traffic due to trust feedback is greatly reduced, making this model relatively lightweight. The complaint mechanism is reserved for actions that are clearly malicious in nature and therefore require the immediate attention of the Sergeant.

Example contexts for which complaints would be warranted include Authorization_Violation, Sensor_Integrity, and Evidence_Tampering. Only complaints that need to receive the immediate attention of the Sergeant are sent directly to the Sergeant. The Sergeant will check the reputation score of the reporting Sentinel as an indicator of its credibility and also check the reputation of the allegedly offending Sentinel. A complaint triggers a policy-based response. If the Sergeant believes the complaint, the Sergeant may remove the offending Sentinel from the Geography and publish the revised Geography to the remaining Sentinels, or it may wait for additional complaints to corroborate the initial complaint, or, if the offending Sentinel recently alerted the Sergeant that it was dealing with problems on its host, the Sergeant may choose to wait for a period of time while Sensors continue to characterize the problem. The latter option reflects the fact that the Sentinels are trusted separately from their hosts.

DualTrust keeps the Sergeant constantly apprised of the global trust situation, first because of the complete trust data stores maintained by the assigned Sentinels which the Sergeant can periodically check (i.e., a pull mechanism) and also because the complaint-based model immediately informs the Sergeant (i.e., a push mechanism) when serious trust breaches occur.

4.5 Trust Evaluation

The context for which we want to measure trustworthiness is the macro-level context of infrastructure defense. Infrastructure defense requires that all entities involved have the highest level of integrity. Any indication of lack of integrity is significant in a security application, regardless of the specific context. Therefore, all trust evidence for a given Sentinel is incorporated into a single reputation score to indicate the Sentinel's integrity in defending the infrastructure. However,

trust evidence is stored according to the context in which it occurred so the contribution of each context to the overall reputation score can be weighted to reflect the importance of that context to the overall score.

Because trust changes over time, only evidence gathered in the last delta time period or in the last n interactions of a given context are included in the calculation. This prevents former good behavior from camouflaging current bad behavior. The choice of a delta time period or n interactions is configurable by the Sergeant for each context.

The reputation score for a Sentinel, S, is calculated as a context(c)-weighted average using only the trust evidence that is newer than a given time delta and was reported by Sentinels that have a reputation score equal to or greater than the reputation threshold specified by the Sergeant. The reputation of a Sentinel, R(S), is the sum, across all contexts, of the number of Passes (vs. Fails) that the Sentinel has received for context c multiplied by the context-specific weight, W, divided by the total number of trust evidence records for the Sentinel for context c, as shown in Fig. 4.

$$R(S) = \sum_{c=1}^{n} \frac{P(S,c) * W(c)}{T(S,c)} \; where \frac{P(S,c)}{T(S,c)} = Threshold \; if \; T(S,c) = 0$$

Fig. 4. Calculating the Reputation Score

The context weights, W(c), have values between 0 and 1; the ΣW(c) must be 1. This equation produces a reputation score normalized to be between 0 and 1. When the system starts, it will take awhile to gather the trust evidence data necessary for this equation. In the meantime, if the number of evidence records received for a given Sentinel and context is 0, then the configured reputation threshold value will be used in place of P(S,c) / T(S,c).

To improve performance, reputation scores can be cached. However, the cached score must be invalidated when new reputation evidence is available or new context weights are distributed by the Sergeant.

4.6 Reputation-Enhanced Trust Relationship Scenario

This section considers how several trust-enhanced Sentinel relationship scenarios would unfold. It considers what trust evidence is checked, how trust evidence is collected, and how and when reputation scores are used.

This section assumes that Sentinel authentication has already occurred and both Sentinels have a Sentinel role-based authorization credential signed by the Sergeant.

Three types of checks are performed in the following order – (1) authorization, (2) any applicable trust evidence-generating checks, and (3) reputation. Trust-evidence generating checks are done prior to calculation of the reputation score so the new evidence can be included in the calculation.

If an authorization violation is found, a complaint is immediately filed with the Sergeant. No remaining authorization checks are required since the authorization checks are ordered such that the broader checks are completed first, and a violation of the first renders the latter of no additional significance. The trust evidence-generating checks and reputation checks are also not performed in this case since the entity whose trustworthiness is being checked is not even legitimate.

Horizontal Trust Relationship 1: A Sensor is passed from the sending Sentinel to the receiving Sentinel. Before sending a Sensor to the receiving Sentinel (a neighbor in the Geography selected by the Sensor), the sending Sentinel will perform the following checks to determine if it can trust the receiving Sentinel. It will proceed to send the Sensor only if merited, and will otherwise require the Sensor to choose a different destination. It does not check the receiving Sentinel's authorization because the Sentinel was selected by the Sensor from the latest Geography received from the Sergeant.

Sender's Check 1: Does the receiving Sentinel have a good reputation? Trust evidence for the receiving Sentinel is gathered and a reputation score is calculated. The sending Sentinel will continue the transfer only if the receiving Sentinel's reputation score is above a minimum threshold determined by the Sergeant's policy.

On the other side of the transaction, the receiving Sentinel, before accepting a Sensor from a sending Sentinel, must consider whether it has sufficient trust in the sending Sentinel. The receiving Sentinel checks the sending Sentinel's current authorization and reputation. If any of the checks fail, the receiving Sentinel will either refuse the Sensor or terminate it upon arrival depending on the Sergeant's policy.

Receiver's Check 1: Is the sending Sentinel a member of the current Geography? If the sending Sentinel is not a member of the current Geography, the receiving Sentinel will file a complaint about the sending Sentinel with the Sergeant and will then either refuse the Sensor or terminate it upon arrival, depending on the Sergeant's policy. The context for the complaint is Former_Member. Authorization findings are not stored as trust evidence because other Sensors are able to perform the same conclusive authorization check using their local copy of the Geography without having to gather trust data or perform a calculation.

Receiver's Check 2: Is the sending Sentinel a neighbor in the current Geography? If the sending Sentinel is not a neighbor in the current Geography, the receiving Sentinel will file a complaint about the sending Sentinel with the Sergeant and will either refuse the Sensor or terminate it upon arrival. The context for the complaint is Not_A_Neighbor.

Receiver's Check 3: Is the cryptographic hash of the Sensor's serialized code the expected hash value? Depending on whether the hash is the expected value, the receiving Sentinel stores the reputation evidence (pass or fail), and depending on policy, may also send a complaint to the Sergeant. In both cases, the context

is Sensor_Integrity. The context weight for Sensor_Integrity should reflect the significance of code integrity.

Receiver's Check 4: Did the sending Sentinel provide adequate resources to the Sensor? The Sensor carries arrival timestamps from its two most recent hosts. Based on the timestamps, the receiving Sentinel calculates whether the lag from when the Sensor was sent to the previous Sentinel to the present time exceeds a configured threshold. This would indicate that the sending Sentinel did not provide adequate resources (CPU cycles, etc.) to the Sensor to allow it to run in a timely fashion. This could be due to an overloaded system or due to maliciousness; therefore, the threshold value should be configured to not regularly punish the Sentinel of a busy system. The threshold is configured and distributed as part of the Sergeant's policy. The receiving Sentinel stores the trust evidence (pass or fail) with the context set to Sensor_Resourcing.

Receiver's Check 5: Does the sending Sentinel have a good reputation? Trust evidence for the sending Sentinel is gathered and a reputation score is calculated. The sending Sentinel's reputation score must be above a minimum threshold determined by the Sergeant's policy.

Horizontal Trust Relationship 2: Hosting Sentinel Runs Sensor Created by the Creating Sentinel. Because the Sensor's hash was checked upon receipt by the receiving Sentinel (called the hosting Sentinel in the context of this trust relationship), we know that the code has not changed and, therefore, any issues with Sensor execution reflect on the creating Sentinel rather than the sending Sentinel. Before allowing the Sensor to execute, the hosting Sentinel considers the creating Sentinel's current authorization and reputation.

Host's Check 1: Is the creating Sentinel a member of the current Geography? If the creating Sentinel, as determined from the Sensor's authorization credential, is not a member of the current Geography, the hosting Sentinel will infer that the creating Sentinel is not trustworthy and will terminate the Sensor. This is not reported to the Sergeant as an authorization violation because the creating Sentinel is not responsible for the fact that the Sensors it created are still active once it has been removed from the Geography. However, if the current Geography was received from the Sergeant prior to the Sensor handoff from the prior Sentinel (based on the timestamps the Sensor carries), the hosting Sentinel knows that the sending Sentinel should have terminated the Sensor. It writes trust evidence for the sending Sentinel (pass or fail) using the context of Sensor_Policing.

Host's Check 2: Does the creating Sentinel have a good reputation? Trust evidence for the creating Sentinel is gathered and a reputation score is calculated. If the score is above a minimum threshold determined by the Sergeant's policy, the executing Sentinel proceeds to run the mobile Sensor agent's code.

Host's Check 3: Did the Sensor perform in a trustworthy manner? After the Sensor code has been run, the Sentinel provides reputation feedback for the creating Sentinel based on factors such as the following:

- Did the attempted or actual privileges of the Sensor exceed what was allowed?
- Did the attempted or actual resource consumption of the Sensor exceed what was allowed?
- Did the Sensor falsely report data that the executing Sentinel knows is not true?
- Did the Sensor disrupt the host in some way?

If the Sensor fails any of these checks, trust evidence is recorded with feedback=Fail; otherwise, feedback is set to Pass. The context is Sensor_Performance. The mechanisms by which these factors are determined are outside the scope of this paper, which simply assumes that such a mechanism exists.

Vertical Trust Relationship: Sergeant Entrusts Sentinel with Enforcing Policy. The Sergeant must be able to depend upon the Sentinels to carry out its policies, so it exercises a continual oversight function much like an employer/employee relationship. If a Sentinel is found to not be in compliance with policy, the Sergeant has the option of removing the Sentinel from the Geography until the problem is resolved The Sergeant has multiple methods by which to determine whether a Sentinel is in compliance with policy:

- The Sentinel's global reputation (based on trust evidence periodically gathered from the Sentinels and calculated) falls below a reputation threshold set by the Sergeant. The Sergeant is also prompted to calculate the global trust of a Sentinel when another Sentinel sends a complaint that its trust has fallen below the allowed threshold (i.e., context is Low_Reputation).
- The Sergeant has received one or more complaints about authorization violations and the reporting Sentinel's reputation score (which indicates its credibility) is above the threshold set by the Sergeant.
- Anomaly detection mechanisms [16] (outside the scope of this paper) identify policy compliance issues.

5 Conclusions and Future Work

This section summarizes the reasons why DualTrust meets the requirements for managing trust (see section 4.2) in a swarm-based autonomic computing system.

DualTrust focuses on the persistent entities of the autonomic computing system, the autonomic managers, because reputation evidence can be gathered over a longer time for persistent entities and is therefore more meaningful than for ephemeral entities. There are also considerably fewer autonomic managers than swarming Sensors, so this focus addresses scalability as well.

Because the evidence storage mechanism stores complete reputation evidence for a given peer replicated across a few selected nodes, it meets the requirement for complete reputation evidence data to be readily available without having to request it from all of the Sentinels. It also enables the Sergeant to have ready access to reputation evidence even though it is not one of the peers. The

replication of the evidence provides for fault tolerance, including tolerance for malicious behavior.

The design of the evidence distribution mechanism minimizes network traffic for scalability. To further improve scalability, an option is presented for having the storing Sentinel calculate and distribute reputation scores rather than reputation evidence.

With the exception of the optional, specialized monitoring agents, the DualTrust model does not constrain adaptation of the Sensor swarm or the individual Sensors in any way. One reason that Sensors can be created and adapted freely is that Sensor trust is handled through trust in the creating Sentinels and through Sensor code integrity rather than requiring the overhead of identification and key pairs for each Sensor.

Additional research is needed to implement DualTrust and to compare the performance of the evidence distribution model and the score distribution model.

Acknowledgments. Much of this research was funded through the Information and Infrastructure Integrity Initiative (I4), an internal (Laboratory Directed Research and Development) investment of the Pacific Northwest National Laboratory in Richland, WA. The Pacific Northwest National Laboratory is managed for the US Department of Energy by Battelle Memorial Institute under Contract DEAC05-76RL01830. Dr. David Bakken and Dr. Ioanna Dionysiou were supported by the National Science Foundation under Grant CNS 05-24695 (CT-CS: Trustworthy Cyber Infrastructure for the Power Grid (TCIP)) and Department of Energy Award Numbers DE-OE0000097[3] (TCIPG) and DE-OE00000321 (GridSim). We appreciate the support of our sponsors.

References

1. Maiden, W.M., Haack, J.N., Fink, G.A., McKinnon, A.D., Fulp, E.W.: Trust Management in Swarm-Based Autonomic Computing Systems. In: Symposia and Workshops on Ubiquitous, Autonomic and Trusted Computing, pp. 46–53 (2009)
2. Dorigo, M., Maniezzo, V., Colorni, A.: The Ant System: Optimization by a Colony of Cooperating Agents. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 26(1), 29–41 (1996)
3. Hoffmeyer, J.: The Swarming Body. In: Semiotics around the world: Synthesis in diversity. Proceedings of the Fifth Congress of the International Association for Semiotic Studies, Berkeley 1994, pp. 937–940. Mouton de Gruyter, Berlin (1997)

[3] Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

4. Kephart, J.O., Chess, D.M.: The Vision of Autonomic Computing. Computer 36, 41–50 (2003)
5. An Architectural Blueprint for Autonomic Computing, 4th edn. Technical report, IBM Corporation, New York (2006), http://www-01.ibm.com/software/tivoli/autonomic/pdfs/AC_Blueprint_White_Paper_4th.pdf
6. Haack, J.N., Fink, G.A., Maiden, W.M., McKinnon, A.D., Fulp, E.W.: Mixed-Initiative Cyber Security: Putting humans in the right loop. In: The First International Workshop on Mixed-Initiative Multiagent Systems (MIMS) (2009), http://u.cs.biu.ac.il/~sarned/MIMS_2009/papers/mims2009_Haack.pdf
7. Wolpert, D., Tumer, K.: An Introduction to Collective Intelligence, Technical Report NASA-ARC-IC-99-63, NASA Ames Research Center, California (1999)
8. Maiden, W.M.: Trust Management Considerations for the Cooperative Infrastructure Defense Framework: Trust Relationships, Evidence, and Decisions, Technical Report PNNL-19117, Pacific Northwest National Laboratory, Richland, Washington (2010), http://www.pnl.gov/main/publications/external/technical_reports/PNNL-19117.pdf
9. Jansen, W., Karygiannis, T.: NIST Special Publication 800-19 – Mobile Agent Security. Technical report, National Institute of Standards and Technology, Gaithersburg, Maryland (1999)
10. Lin, K.-J., Lu, H., Yu, T., Tai, C.: A Reputation and Trust Management Broker Framework for Web Applications. In: Proceedings of the 2005 IEEE International Conference on e-Technology, e-Commerce, and e-Service, pp. 262–269 (2005)
11. Lin, C., Varadharajan, V., Wang, Y., Pruthi, V.: Trust Enhanced Security for Mobile Agents. In: Seventh IEEE International Conference on E-Commerce Technology, pp. 231–238 (2005)
12. Tan, H.K., Moreau, L.: Certificates for Mobile Code Security. In: Proceedings of the 2002 ACM Symposium on Applied Computing, pp. 76–81. ACM, New York (2002)
13. Lin, C., Varadharajan, V., Wang, Y., Pruthi, V.: Security and Trust Management in Mobile Agents: A New Perspective. In: 2nd International Conference on Mobile Technology, Applications and Systems, pp. 1–9 (2005)
14. Lin, C., Varadharajan, V.: Trust Enhanced Security - A New Philosophy for Secure Collaboration of Mobile Agents. In: International Conference on Collaborative Computing: Networking, Applications and Worksharing, pp. 1–8 (2006)
15. Xiong, L., Liu, L.: PeerTrust: Supporting Reputation-Based Trust for Peer-to-Peer Electronic Communities. IEEE Transactions on Knowledge and Data Engineering 16, 843–857 (2004)
16. Stakhanova, N., Basu, S., Wong, J., Stakhanov, O.: Trust Framework for P2P Networks Using Peer-Profile Based Anomaly Technique. In: 25th IEEE International Conference on Distributed Computing Systems Workshops, pp. 203–209 (2005)
17. SPKI/SDSI Certificates, http://world.std.com/~cme/html/spki.html
18. Better Business Bureau, http://www.bbb.org/us/
19. Aberer, K., Despotovic, Z.: Managing Trust in a Peer-2-Peer Information System. In: Proceedings of the Tenth International Conference on Information and Knowledge Management, pp. 310–317. ACM, New York (2001)

MIRAGE: A Management Tool for the Analysis and Deployment of Network Security Policies

Joaquin Garcia-Alfaro, Frédéric Cuppens,
Nora Cuppens-Boulahia, and Stere Preda

Institut Télécom, Télécom Bretagne,
CS 17607, 35576 Cesson-Sévigné, France
joaquin.garcia-alfaro@acm.org,
forename.surname@telecom-bretagne.eu

Abstract. We present the core functionality of MIRAGE, a management tool for the analysis and deployment of configuration policies over network security components, such as firewalls, intrusion detection systems, and VPN routers. We review the two main functionalities embedded in our current prototype: (1) a bottom-up analysis of already deployed network security configurations and (2) a top-down refinement of global policies into network security component configurations. In both cases, MIRAGE provides intra-component analysis to detect inconsistencies in single component deployments; and inter-component analysis, to detect multi-component deployments which are not consistent. MIRAGE also manages the description of the security architecture topology, to guarantee the proper execution of all the processes.

Keywords: Network security, Access control, Analysis of configurations, OrBAC, Policy refinement.

1 Introduction

Despite the advances in the field of network security technologies, such as filtering of traffic, use of encrypted communications, and deployment of authentication mechanisms, there may always be errors or flaws that can be exploited by unauthorized parties. The use of firewalls, NIDSs (network intrusion detection systems), and VPN (Virtual Private Network) routers, is still the dominant method to survey and guarantee the security policy in current corporate networks. The configuration of these components is based on the distribution of security rules that state what is permitted and what is prohibited in a system during normal operations. This configuration must be consistent, addressing the same decisions under equivalent conditions, and not repeating the same actions more than once. Otherwise, the existence of anomalies in their configuration rules may lead to weak security policies (potentially easy to be evaded by unauthorized parties). The update of the component configurations can also introduce new anomalies. There is, therefore, a clear need of support tools to guide the operators when performing such tasks.

Our research work has studied the combination of two main strategies in order to manage this problem. The first strategy is the use of an audit mechanism that analyzes

J. Garcia-Alfaro et al. (Eds.): DPM 2010 and SETOP 2010, LNCS 6514, pp. 203–215, 2011.

already deployed configurations, signals inconsistencies, and yields consistent configurations. Through this mechanism, moreover, we can fold existing policies and create a consistent and global set of rules — easy to maintain and manage by using a single syntax [13,14]. The second strategy is the use of a refinement mechanism that guarantees the proper deployment of such rules into new systems, yet free of inconsistencies. These two research strategies have been implemented into a software prototype called MIRAGE (which stands for MIsconfiguRAtion manaGEr).

Developed as a web service, MIRAGE can autonomously be executed under the control of several operators in order to offer the system with the following functions: (1) an intra-component analysis which detects inconsistencies between rules within single security component policies [12,10]; (2) an inter-component analysis of rules to detect inconsistencies between configurations of different devices [15]; (3) an aggregation mechanism to fold all the existing policies into a single, and consistent, global set of rules [14]; and (4) a refinement process to properly deploy the global set of rules over new security components [17,18]. In all four cases, MIRAGE utilizes a description of the topology of the whole security architecture.

The use of MIRAGE can highly benefit the maintenance of multihomed autonomous systems. It might remain connected to the network and provide assistance to the complete set of component in the event of configuration maintenance or redeployment of configuration to face events such as detection of malicious activity or failures. In this sense, the refinement mechanism offered by MIRAGE guarantees that the set of rules deployed over the different components of a system is always consistent, not redundant, and optimal [19].

Paper Organization — Section 2 reviews recent results implemented in the MIRAGE prototype for guaranteeing correctness and consistency on single and distributed network security policies. Section 3 compares these approaches implemented in MIRAGE with other solutions proposed in both the science and the industry community. Section 4 closes the paper.

2 MIRAGE Prototype

MIRAGE is a management tool for guaranteeing the correctness and the consistency of configuration rules on single and distributed network security policies. It implements an analysis of components' configurations (i.e., configurations of firewalls, NIDSs, and VPN routers) to detect anomalies on their deployment. To do so, MIRAGE implements four main functions: intra-component analysis for the detection of inconsistencies between configuration rules within single security component policies; inter-component analysis of rules to detect inconsistencies between configurations of different devices; aggregation of policies for the creation of a consistent and global set of rules; and refinement mechanism for the deployment of global policies over the different components of new systems. We address in the sequel the key aspects of some of these functionalities implemented in the current version of our prototype.

2.1 Bottom-Up Analysis of Network Configurations

We assume here that a security policy has been empirically deployed into the network based on security administrator expertise and flair. It is then advisable to analyze the security rules deployed to detect and correct policy inconsistencies. These inconsistencies are often the origin of security holes exploited by dishonest parties. MIRAGE addresses this process and provides a discovery of inconsistencies and redundancies from component configurations. This process is presented based on two different schemes: (1) single- and (2) multi-component analysis.

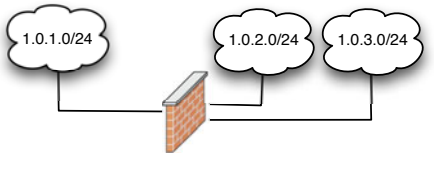
Single-component Analysis

MIRAGE provides a deterministic process to detect inconsistencies in the configuration of security components. It considers that these devices are configured in a standalone manner, using a set of configuration rules (e.g., filtering rules in the case of a firewall; and alerting rules in the case of a NIDS). A general configuration rule is defined as follows:

$$R_i : \{condition_i\} \rightarrow decision_i \tag{1}$$

Regarding the previous expression, i is the relative position of a rule in the set, $\{condition_i\}$ is a conjunctive set of condition attributes such that $\{condition_i\}$ equals $A_1 \wedge A_2 \wedge ... \wedge A_p$ – being p the number of attributes of the given rule – and $decision$ is a boolean value in $\{true, false\}$. For example, the decision of a filtering rule is positive ($true$) when it applies to a specific value related to *deny* the traffic it matches; and negative ($false$) when it points to *accept* the traffic it matches. Similarly, the decision field of an alerting rule is positive ($true$) when it applies to a specific value related to *alert* about the traffic it matches; and negative ($false$) when it applies to a specific value related to *ignore* the traffic it matches. Based on the sample scenario depicted by Figure 1, and its associated set of rules, we define the following set of anomalies detected by the intra-component audit process.

- *Intra-component Shadowing* — A configuration rule R_i is shadowed in a set of configuration rules R when such a rule never applies because all the packets that R_i may match, are previously matched by another rule, or combination of rules, with higher priority. E.g., rule R_6 is shadowed by the overlapping of rules $R_3 \cup R_5$.
- *Intra-component Redundancy* — A configuration rule R_i is redundant in a set of configuration rules R when the following conditions hold: (1) R_i is not shadowed



R1: {tcp,1.0.1.[10,20]:any, 1.0.1.[50,60]:any} → false
R2: {tcp,1.0.2.[10,255]:any, 1.0.3.[0,255]:any} → false
R3: {tcp,1.0.2.[1,30]:any, 1.0.1.[20,45]:any} → true
R4: {tcp,1.0.2.[20,60]:any, 1.0.1.[25,35]:any} → false
R5: {tcp,1.0.2.[30,70]:any, 1.0.1.[20,45]:any} → false
R6: {tcp,1.0.2.[15,45]:any, 1.0.1.[25,30]:any} → true

Fig. 1. Single filtering policy scenario

by any other rule or set of rules; (2) when removing R_i from R, the security policy does not change. E.g., rule R_4 is redundant to $R_3 \cup R_5$.

– *Intra-component Irrelevance* — A configuration rule R_i is irrelevant when (1) source and destination addresses are within the same zone or (2) the component does not appear in the minimal route that connects the source zone (i.e., the rule matches traffic that never reaches the component). E.g., rule R_1 is irrelevant since both source and destination are in network $1.0.1.0/24$. Rule R_2 is also irrelevant since the filtering device is not part of the minimal route between networks $1.0.2.0/24$ and $1.0.3.0/24$.

The reader can find in [10,15] the algorithms that enable MIRAGE the detection of the inconsistencies presented in this section, as well as correctness and computational complexity of the algorithms. Although we show that the theoretical complexity of the algorithms is very high, we show with a series of experimentations (cf. [15], Section 6) that we are always very far from the worst case. Indeed, only few attributes, such as source and destination addresses, may significantly overlap and exercice a bad influence on the algorithms complexity. Other attributes, such as the protocol or the port numbers, are generally equal or completely different when combining configuration rules. Moreover, when anomalies are discovered, some rules are removed – which significantly reduces the algorithms complexity.

Multi-component Analysis

MIRAGE provides a second audit process to analyze multi-component setups (e.g., distributed architectures with firewalls and NIDSs in charge of multiple network security policies). In this sense, it can assume, for instance, that the role for detecting and preventing network attacks is assigned to several components. It will, then, look for inconsistencies hidden in their configurations. The detection process is based on the similarity between the parameters of configuration rules such as filtering an alerting rules. It checks, indeed, if there are errors in the configurations by comparing the policy deployment over each component that matches the same traffic. Based on the sample scenario depicted by Figure 2, we show in the sequel an example of the kind of inconsistencies detected by the inter-component audit process of MIRAGE.

– *Inter-component Shadowing* — A shadowing anomaly occurs between two components when the following conditions hold: (1) The component that is located closest to the origin of the traffic is a filtering device (e.g., a firewall); (2) The component where the anomaly is detected does not block or report (completely or partially) traffic that is blocked (explicitly, by means of positive rules; or implicitly, by means of its default policy), by the first component in the path (closest to the source). The following table shows some examples.

Rules	**Anomaly**
$C_6\{R_7\} \cup C_6\{R_8\}$ shadows $C_3\{R_1\}$	*full shadowing*
$C_6\{R_8\}$ partially shadows $C_3\{R_2\}$	*explicit partial shadowing*
Close policy of C_2 shadows $C_1\{R_5\}$	*implicit full shadowing*

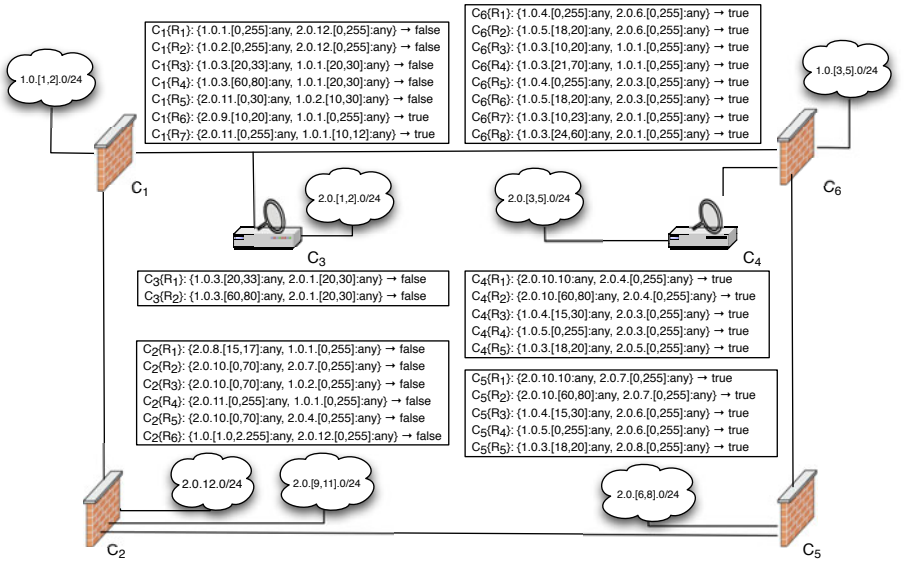


Fig. 2. Example of an inter-component setup

– *Inter-component Redundancy* — A redundancy anomaly occurs between two components when the following conditions hold: (1) The component that is located closest to the origin of the traffic is a filtering device (e.g., a firewall); (2) The component where the anomaly is detected, blocks or reports (completely or partially) traffic that is already blocked by the first component. This kind of redundancy is often introduced by network officers expressly. It is important, however to alert about it, to warn the administrator that the rule has a special meaning (e.g., a message warning that if the rule applies, the upstream filtering devices are not working properly). The following table shows some examples.

Rules	Anomaly
$C_6\{R_1\}$ is redundant to $C_5\{R_3\}$	*full redundancy*
$C_6\{R_5\}$ is redundant to $C_4\{R_3\}$	*full redundancy*
$C_6\{R_2\}$ is redundant to $C_5\{R_4\}$	*partial redundancy*
$C_6\{R_6\}$ is redundant to $C_4\{R_4\}$	*partial redundancy*

– *Inter-component Misconnection* — A misconnection anomaly occurs between two components when the first one, located closest to the source, is a firewall that permits (explicitly, by means of negative rules; or implicitly, through its default policy) all the traffic, or just a part of it, that is then denied by the component where the anomaly is detected. The following table shows some examples.

Rules	Anomaly
$C_5\{R_1\}$ and $C_2\{R_2\}$ are misconnected	*full explicit misconnection*
$C_5\{R_2\}$ and $C_2\{R_2\}$ are misconnected	*partial explicit misconnection*
$C_1\{R_5\}$ and policy of C_2 are misconnected	*full implicit misconnection*
$C_1\{R_6\}$, $C_2\{R_1\}$, and policy of C_2	*partial implicit misconnection*

The reader can find in [15] the algorithms that enable MIRAGE the detection of the inconsistencies presented in this section, as well as correctness profs, computational complexity, and experimental results. The complete set of analyzed configuration can be aggregated into a single, and consistent, global set of rules by using the aggregation mechanism presented in [14]. This global policy is, in fact, the main source of information used by the refinement mechanism presented in the sequel.

2.2 Top-Down Refinement of Global Policies

A second approach to address the management of consistency and correctness of network policies is the use of refinement mechanisms. In this way, we can perform a downward deployment of rules by unfolding a global set of security policies into the configurations of several components and guaranteeing that the deployed configurations are free of anomalies. In [9], for example, we presented a refinement mechanism that uses a formal model for the generation of filtering rules by transforming general rules into specific configuration rules. We address in this section some functionalities addressed by MIRAGE in this sense.

Model-driven Policy Deployment. If manually carried out, the process of deploying network security policies is often errorprone. In fact, without the right structural knowledge of the policy, the deployment of conflicting security requirements becomes very likely. This highlights the necessity of a more structured policy expression, i.e., only a formalized expression of the security policy may guarantee an error-free security policy to be deployed, with no ambiguities, no inconsistencies, no redundancies and no unnecessary details. Thus MIRAGE considers that an access control model and a formalized security policy is the first step toward enforcing by refinement the security of the system.

MIRAGE takes full advantage of the OrBAC model (Organization Based Access Control) [1] which is an extension of RBAC [20]. OrBAC presents a high abstraction level and covers a large panel of security policies since it natively provides means to express both static requirements (i.e., they are enforced once and for all) and *contextual* requirements (i.e., dynamic requirements). The OrBAC notions of *role*, *activity*, and *view* and also *context* prove very useful: the complexity of both the system (tens of firewalls, NIDSs, and VPN routers) and the policy (static and dynamic security requirements) is no longer an issue. The *role* regroups *subjects* (concrete network entities), the *activity – actions* (network services), the *view – objects* (e.g., IP packets, network entities) on which the same rules apply respectively. The notion of *context* confers the possibility to address a larger variety and also a finer granularity of security requirements, it

captures the *conditions* (e.g., environmental factors) in which the security requirements are enforced and met. The OrBAC *context* allows the specification of these conditions directly at an abstract policy level.

The policy refinement mechanism of MIRAGE is in fact a set of deployment algorithms which constitutes the *downward* process: an OrBAC error-free security policy is refined into packages of rules for each security device in the given network. The aim is the *correct* deployment of a security policy, and this is achievable if some specific *security properties* are verified at the end. Such properties guarantee that no intra- nor inter- component anomalies are introduced (cf. Section 2.1). The formal frame to design the refinement mechanism of MIRAGE is presented in [18]. The policy deployment algorithms are developed using the B Method [2], a theorem proving method. The B Method offers the means to cope with the issue of stating the *interesting* security properties: besides an appropriate modeling language for both the OrBAC policy and the system/network specifications, it allows a formal expression of the properties related to the management of the intra- and inter- configuration anomalies during the *downward* process. This is ensured by some B *invariants*. Thus, from the early stage of their B development (i.e., abstract B specification), the policy deployment algorithms of MIRAGE target the interesting security properties. Examples of security properties we took into account, and expressed as B invariants, in [18] are:

- *Completeness* — This property states that if the network path from a *subject* to an *object* is correctly computed (i.e., it exists and the security components belonging to this path have the right functionalities with respect to the current $context_i$) the security OrBAC rule *Is_permitted*(*subject*, *action*, *object*, $context_i$) may and will be deployed. Clearly, this property is closely related to the network's architecture and the assumption of connectedness in the network architecture is required.
- *All traffic are regulated by filtering components* — This property is verified if there is, at least, exactly one firewall or one IPS on the path between the current subject and object.
- *Integrity and confidentiality* — These two properties are related to the establishment of VPN tunnels. The verification starts at higher levels: the current OrBAC security rule should be defined with a *protected* context — meaning that the trafic filtered by the associated rules must be protected by the VPN tunnels. Then, if a path is computed between the subject and the object and a VPN tunnel can be established on this path, the integrity and confidentiality properties are verified.

The work in [18] presents a complete analysis of security properties. Some may be specified at higher levels [8] and some may be identified from specific security requirements. The MIRAGE deployment algorithms were formally proved with the assumption of a conflict-free OrBAC policy and of a correct system architecture, i.e., no lack of security functionalities in the security components placed on the shortest-paths. Hence, as long as the system embeds all necessary security functionalities, there are no concerns in deploying the policies. The refinement provided by MIRAGE is a certified algorithm for

a reliable and automatic security policy deployment. It is, however, realistic to consider that sometimes the system lacks some necessary security functionalities. Our proposed solutions to this problem are presented in the sequel.

Context Aware Policy Deployment. The security policies become more and more contextual. (Re)deploying a contextual security policy depends on the security device functionalities: either (1) the devices include all functionalities necessary to handle a context and the policy is consequently deployed to ensure its automatic changes or (2) the devices do not have the right functionalities to interpret a contextual requirement in its entirety. MIRAGE proposes two solutions to cope with the issue of the (re)deployment of access control policies in a system that lacks the necessary functionalities to deal with contexts:

1. *Dynamic deployment*: MIRAGE considers a central entity (hereafter called *PDP* – Policy Decision Point) which (partially) manages some contexts.
2. *Optimization deployment*: if the previous solution does not stand.

Obviously, the OrBAC formalism is maintained. These two solutions presented in [17] and [19] respectively can then be jointly used whenever the security devices (hereafter called *PEPs* – Policy Enforcement Points) are not rich enough in functionalities so as to manage all contexts by themselves. In this way, a complete deployment of the (contextual) policy may be achieved.

The Methodology. Let SR = (Decision, Role, Activity, View, Ctx) be a security rule of the OrBAC policy P (SR $\in P$) and SR$'$ = (Decision, Role, Activity, View, Ctx$'$) with Decision $\in$ {Permission, Prohibition}. We call SR$'$ the SR *contextual version* over the context Ctx$'$. Let PEP$_i$ be an enforcement point able to manage only the context Ctx$'$ and SR be the security rule PEP$_i$ must enforce. We investigate how SR$'$ can be deployed and thus enforced by PEP$_i$ even if Ctx$'$ is not equal to the context Ctx of the initial rule SR to be deployed. The final aim is to deploy the SR rule and one of the following situations appears:

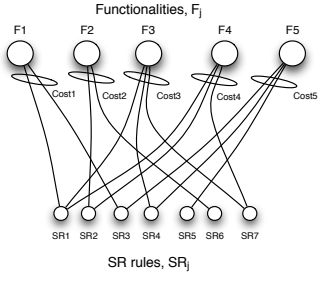
- *Case 1* — The PEP$_i$ manages the entire Ctx context. The rule SR is directly deployed over PEP$_i$ and the PDP does not manage SR anymore. Otherwise, the PDP has to manage a part of the Ctx context.
- *Case 2* — The PEP$_i$ manages only Ctx$_2$, a part of the Ctx context. We note this case as Ctx$_1$ = Ctx - Ctx$_2$. The deployment is *dynamic* and the PDP manages Ctx$_1$: the PDP must deploy SR$'$, the SR contextual version over Ctx$_2$ on the PEP$_i$ when Ctx$_1$ becomes active. Once Ctx$_1$ is deactivated, the PDP must be able to retrieve the deployed SR$'$ from PEP$_i$. For example and as suggested in [17], Ctx$_1$ may represent a threat context activated by the detection of some intrusion. Thus, *Case 2* provides means to dynamically redeploy the policy to face this intrusion. This represents the *dynamic deployment* solution.
- *Case 3* — Neither the PEP$_i$, nor the PEP$_i$ and the PDP working together manage the Ctx context. Then, two solutions are possible: (I) there may be a context closely related with Ctx which is still managed by the system as described in *Case 2* (in fact,

we refer to the OrBAC context *hierarchies*); or (II) if the system does not provide *hierarchies*, the last option is to find a security functionality closely-equivalent to the one necessary to handle the Ctx context. Both (I) and (II) represent the *best deployed policy* solution.

Dynamic Deployment. The formalization of this solution is based on the use of *ECA* (*Event Condition Action*) [5]. The algorithms running at the PDP level deploy (or retrieve) security rules over the PEPs when the contexts are activated (or deactivated). To be effective, this solution requires a specific communication protocol between the PDP and the PEPs. Actually our method uses the Netconf (cf. http://www.ops.ietf.org/netconf/) protocol with the Yencap (cf. http://ensuite.sourceforge.net/) open-source implementation which we adapted accordingly. Several performance tests were realized and presented in [17]. The results proved to be satisfactory.

Best Deployed Policy. There are scenarios in which the PDP and/or PEPs cannot entirely handle the Ctx context related to an SR rule. Instead of skipping such rules (with the result of, for example, a too restrictive deployed policy at the end), MIRAGE proposes the following solutions of: (1) finding a closely related context to the unmanaged one and which may be managed by the PDP and/or PEP and/or (2) finding a close enough functionality to deal with the unmanaged context if solution (1) does not apply. OrBAC proves very effective for the first solution since OrBAC provides *context hierarchies*. Thus, it is enough to find either the *more specialized* context than Ctx (to deploy permissions) or less specialized ones (to deploy prohibitions).

The second solution is solved with an optimization approach. The system presents no optimal functionality to manage the Ctx context but only *closely-equivalent* ones. We declare these functionalities with the *Close_Fs*() predicate. A notion of *cost* of using a given functionality to deploy certain rules in the Ctx context is introduced and the deployment problem is transformed into an optimization one. The result is a bipartite graph where the optimization solution is obtained with linear programming. Figure 3 depicts a proper example. Notice that, globally, the cost of deploying the SR rules over the Ctx contexts is minimized. The optimal functionalities necessary to handle the Ctx context will be substituted by closely-equivalent ones.



(a) The LP Problem.

Case A

minimize $\sum_{j=1}^{n} \text{Cost}_j x_j$,

subject to $\sum_{j=1}^{n} d_{ij} x_j \geq 1,\ i \in (1...m)$

$x_j = 0$ or $1,\ j \in (1...n)$

where $x_j = 1$ if j is in FS, $x_j = 0$ otherwise; and $d_{ij} = 1$ if SR_i could be deployed using the functionality F_j, $d_{ij} = 0$ otherwise.

Case B

minimize $\sum_{j=1}^{n} \text{Cost}_j x_j$,

subject to $\sum_{j=1}^{n} d_{ij} x_j = 1,\ i \in (1...m)$

$x_j = 0$ or $1,\ j \in (1...n)$

where $x_j = 1$ if j is in FS, $x_j = 0$ otherwise; and $d_{ij} = 1$ if SR_i could be deployed using the functionality F_j, $d_{ij} = 0$ otherwise.

(b) The LP Solutions.

Fig. 3. LP Problem and Solutions

3 Related Works

A significant amount of work has been reported in the area of security management at network level in order to analyze and fix existing configurations. The most significant approach to firewall policy analysis is the one by Al-Shaer et al. (e.g., approaches presented in [3,4]) which provides efficient solutions to detect policy anomalies in both single- and multi-firewall configuration setups. Their detection algorithms are based on the analysis of relationships between rules two by two. Therefore, errors due to the union of rules are not explicitly considered (as our approach does). Some workarounds can be provided to solve this situation. For instance, it is possible to break down the initial set of rules into an equivalent set of rules free of overlaps between rules. However, no specific algorithms were provided in [3,4] to manage this solution. Another related work is the proposal presented in [21], which uses a model checking formalism for detecting inconsistencies and redundancies in single firewall configurations. This proposal handles the limitation pointed out in the works by Al-Shaer et al., by addressing directly the way how traffic is handled by the components. The complete set is divided in three main sets: traffic that is permitted, traffic that is prohibited, and traffic for which no rules apply. The proposal in [21], as well as other similar approaches, such as [16], only address intra-component analysis. Moreover, none of them have presented specific mechanisms for verifying policies other than filtering ones.

Regarding the analysis of VPN routers' configurations, the most significant approach compared to ours is proposed in [11]. The authors propose a technique that simulates VPN tunneling processing and reports any violation of the security policy requirements. In their approach, if an access rule concerning a protected traffic between two points is implemented by configuring more than one VPN overlapping tunnel, the risk is that in some network zones the IP packets circulate without any protection. The authors present a discovery process to detect such situations and propose a high-level language to deal with VPN policies. Although this approach can discover some violations in a certain simulation scenario, there is no guarantee that it discovers every possible violation that may exist. In addition, the proposed technique only discovers VPN conflicts resulting from incorrect tunnel overlap, but does not address the other types of conflicts. Another attempt to deploy VPN configurations free of anomalies is [6] where the authors propose a central-entity approach with high level language conflict resolution techniques also - the algorithms remained however unevaluated. However, a significant aspect is ignored in both previous approaches: the security policy cannot be seen as two independent sets of requirements (i.e., VPN tunnels and firewalls modeled separately). The use of a single access control model in our approach solves this limitation and allows us to deal with a global set of security requirements and to address inter-mechanisms anomalies (e.g., firewall vs. VPN conflicts) at the same time.

Another significant approach compared to ours is the RBAC-based proposal presented in [7], called Firmato. This new solution aims at configuring filtering devices following an approach of separation between the security policy model and the technology specifications. It, therefore, ensures policy deployment independently of the network topology. The tool is based on an *Entity — Association* model (abstract level) which takes into account the network topology as a role. The instantiation of the model is based on a specific language that allows a downward transformation of the global

policy into a set of firewall configurations. However, the use of the role concept used in Firmato, which defines the network capabilities, becomes ambiguous in its semantics. The authors use the notion of *group* to handle this situation. A group can identify, in fact, a set of hosts but also a role or a set of roles. Its use does not ensure, indeed, a clear separation between the network level and the security policy, making difficult the use of this tool to model complex networks. The authors use, moreover, privilege inheritance through group hierarchies in order to derive permissions. If permission inheritance is related to the so-called *open group*, prohibitions are inherited through a *close group*. The notion of group clearly introduces ambiguities and seems to be useless at this abstraction level.

Support tools can also be used to assist administrators in their task of configuring security devices. Proper examples are the LogLogic Security Change Manager (for more info, cf. http://loglogic.com/products/), formerly known as Solsoft Policy Server and Network Security Policy Server, the Firewall Builder (cf. http://fwbuilder.org/), CheckPoint SmartCenter (cf. http://checkpoint.com/products/smartcenter/), Juniper Network and Security Manager (cf. http://www.netutils.com/), as well as the Cisco Security Manager (cf. http://cisco.com/go/csmanager). In a relatively high level language, these support tools allow the configuration of different vendors' devices and support the security administrators in the deployment of large configurations on heterogeneous networks. We observe the following problems when using such tools. First, they do not offer a semantic model rich enough to express a global security policy. Although it is possible to define variables, and thus to define access rules involving such variables, the administration tasks are not much simplified. The security officer always needs a global view of the topology in order to correctly assign each rule to network devices; then, there is no automatic discovery of security devices that optimally implement an access rule involving an IP source and a destination. Furthermore, the lack of a real downward approach like ours is partially replaced by other tools (e.g., Cisco conflict discovery tools) that need the security officer's assistance and that unfortunately only guarantee conflict resolution for local configurations.

4 Conclusions

We addressed the managing of network security policies free of anomalies or inconsistencies. Two main approaches were presented: (1) the use of bottom-up process to detect and fix configuration errors over components already deployed; and (2) the use of a top-down process to perform an automatic deployment of component configurations free of inconsistencies. The implementation of these two approaches in a software prototype demonstrates the practicability of our work. We finally compared the functionality of MIRAGE with some other solutions proposed in both the science and the industry community, and showed some advantages of our approaches. As future work, it is expected to add new a feature in MIRAGE to manage the update of components' configurations. This new feature will guide the operators to determine the impact that the removal or the addition of new configuration rules in the system might suppose. It is also expected to give support to determine dynamic tuning of configurations. In this case, the new feature is expected to compare and test the equivalence between different

configurations. For example, the security operator can verify whether the new settings of a new configuration setup will perform well enough, and in compliance with the global security policy. Finally, it is also intended to complement the upward and the downward approaches offered by MIRAGE with an automatic discovery of roles associated with different security components already deployed in the system. It is planned the use of role mining techniques, for example, to analyze existing access control roles associated to the components (to derive, after the analysis, the appropriate rules of the global configuration).

Acknowledgments. This work has been supported by a grant from the Brittany region of France and by the following projects: POLUX ANR-06-SETIN-012, SEC6 Project, TSI2007-65406-C03-03 E-AEGIS, and CONSOLIDER CSD2007-00004 "ARES".

References

1. Abou el Kalam, A., Baida, R., Balbiani, P., Benferhat, S., Cuppens, F., Deswarte, Y., Miège, A., Saurel, C., Trouessin, G.: Organization Based Access Control. In: IEEE 4th Intl. Workshop on Policies for Distributed Systems and Networks, Lake Come, Italy, pp. 120–131 (2003)
2. Abrial, J.R.: The B-Book — Assigning Programs to Meanings. Cambridge University Press, Cambridge (1996) ISBN 052149619-5
3. Al-Shaer, E.S., Hamed, H.H.: Discovery of Policy Anomalies in Distributed Firewalls.. In: IEEE INFOCOM 2004 (March 2004)
4. Al-Shaer, E.S., Hamed, H.H.: Taxonomy of Conflicts in Network Security Policies. IEEE Communications Magazine 44(3) (March 2006)
5. Baral, C., Lobo, J., Trajcevski, G.: Formal Characterization of Active Databases. In: Bry, F. (ed.) DOOD 1997. LNCS, vol. 1341. Springer, Heidelberg (1997)
6. Baek, S., Jeong, M., Park, J., Chung, T.: Policy based Hybrid Management Architecture for IP-based VPN. In: Network Operations and Management Symposium, NOMS 2000 (2000)
7. Bartal, Y., Mayer, A., Nissim, K., Wool, A.: Firmato: A Novel Firewall Management Toolkit. In: IEEE Symposium on Security and Privacy, Oakland, California, pp. 17–31 (May 1999)
8. Benaïssa, N., Cansell, D., Méry, D.: Integration of Security Policy into System Modeling. In: Julliand, J., Kouchnarenko, O. (eds.) B 2007. LNCS, vol. 4355, pp. 232–247. Springer, Heidelberg (2006)
9. Cuppens, F., Cuppens, N., Sans, T., Miège, A.: A formal approach to specify and deploy a network security policy. In: Second Workshop on Formal Aspects in Security and Trust, Toulouse, France, pp. 203–218 (August 2004)
10. Cuppens, F., Cuppens, N., Garcia-Alfaro, J.: Misconfiguration management of network security components. In: 7th International Symposium on System and Information Security (SSI 2005), Sao Paulo, Brazil, pp. 1–10 (November 2005)
11. Fu, Z., Wu, S., Huang, H., Loh, K., Gong, F., Baldine, I., Xu, C.: IPSec/VPN Security Policy: Correctness, Conflict Detection and Resolution. In: International Policy Workshop (January 2001)
12. García-Alfaro, J., Cuppens, F., Cuppens-Boulahia, N.: Towards Filtering and Alerting Rule Rewriting on Single-Component Policies. In: Górski, J. (ed.) SAFECOMP 2006. LNCS, vol. 4166, pp. 182–194. Springer, Heidelberg (2006)
13. García-Alfaro, J., Cuppens, F., Cuppens-Boulahia, N.: Analysis of policy anomalies on distributed network security setups. In: Gollmann, D., Meier, J., Sabelfeld, A. (eds.) ESORICS 2006. LNCS, vol. 4189, pp. 496–511. Springer, Heidelberg (2006)

14. Garcia-Alfaro, J., Cuppens, F., Cuppens, N.: Aggregating and Deploying Network Access Control Policies. In: 2nd International Conference on Availability, Reliability and Security (ARES 2007), Vienna, Austria, pp. 532–539. IEEE Computer Society, Los Alamitos (April 2007)
15. Garcia-Alfaro, J., Cuppens, F., Cuppens, N.: Complete Analysis of Configuration Rules to Guarantee Reliable Network Security Policies. International Journal of Information Security 7(2), 103–122 (2008)
16. Liu, A.X., Gouda, M.G.: Complete Redundancy Detection in Firewalls. In: 19th Annual IFIP Conference on Data and Applications Security (DBSec 2005), Storrs, Connecticut, pp. 196–209 (August 2005)
17. Preda, S., Cuppens, F., Cuppens-Boulahia, N., Garcia-Alfaro, J., Toutain, L., Elrakaiby, Y.: A Semantic Context Aware Security Policy Deployment. In: ACM Symposium on Information, Computer and Communications Security, Sydney, Australia, pp. 251–261 (March 2009)
18. Preda, S., Cuppens-Boulahia, N., Cuppens, F., Garcia-Alfaro, J., Toutain, L.: Model-Driven Security Policy Deployment: Property Oriented Approach. In: Massacci, F., Wallach, D., Zannone, N. (eds.) ESSoS 2010. LNCS, vol. 5965, pp. 123–139. Springer, Heidelberg (2010)
19. Preda, S., Cuppens-Boulahia, N., Cuppens, F., Toutain, L.: Architecture-Aware Adaptive Deployment of Contextual Security Policies. In: Fifth International Conference on Availability, Reliability and Security (ARES 2010). IEEE Computer Society, Los Alamitos (February 2010)
20. Sandhu, R., Coyne, E., Feinstein, H., Youman, C.: Role-Based Access Control Models. IEEE Computer 29(2), 38–47 (1996)
21. Yuan, L., Mai, J., Su, S., Chen, H., Chuah, C., Mohapatra, P.: FIREMAN: a toolkit for FIREwall Modeling and ANalysis. In: IEEE Symposium on Security and Privacy, Oakland, California, pp. 199–213 (2006)

A DSL for Specifying Autonomic Security Management Strategies*

Ruan He[1], Marc Lacoste[1], Jacques Pulou[1], and Jean Leneutre[2]

[1] Orange Labs, France
{ruan.he,marc.lacoste,jacques.pulou}@orange-ftgroup.com
[2] Telecom ParisTech, France
jean.leneutre@telecom-paristech.fr

Abstract. Existing self-protection frameworks so far hardly addressed the specification of autonomic security adaptation strategies which guide risk-aware selection or reconfiguration of security mechanisms. *Domain-Specific Languages (DSL)* present many benefits to achieve this goal in terms of simplicity, automated strategy verification, and run-time integration. This paper presents a DSL to describe security adaptation policies. The DSL is based on the condition-action approach and on a taxonomy of threats and applicable reactions. The DSL also allows to capture trade-offs between security and other concerns such as energy efficiency during the decision making phase. A translation mechanism to refine the DSL into a run-time representation, and integrate adaptation policies within legacy self-protection frameworks is also presented.

1 Introduction

Advances in wireless communications and embedded systems turned pervasive networking into a reality. Those environments are more open, mobile, and dynamic than traditional distributed systems. At the same time, they unravel a whole new landscape of rapidly changing threats and of heterogenous security requirements, calling for strong and yet highly flexible security mechanisms. In such settings, managing protection "by-hand" becomes far too complex. Thus, the *autonomic approach* to security management [8] was a major step forward to address those issues, a system now being able to protect itself without or with minimal human intervention.

Existing self-protection frameworks [15,9] realize autonomic control loops to manage protection mechanisms, applying some guiding strategies called *adaptation policies*. However, those frameworks assume the policies to be already defined, and how to specify them explicitly is usually not addressed. Furthermore, trade-offs between different concerns such as security, QoS, or energy efficiency are not taken into account. Adaptation strategies are usually specified in ad hoc formalisms such as if-then-else rules, which are not intuitive for users, and lack error-checking support mechanisms. Another aspect also missing is how to easily integrate such policies into existing security management frameworks.

* This work has been partially funded by the ANR SelfXL project.

J. Garcia-Alfaro et al. (Eds.): DPM 2010 and SETOP 2010, LNCS 6514, pp. 216–230, 2011.

A *Domain-Specific Language (DSL)* is a high-level language devoted to explicit different perspectives of a given domain, providing specific language support to describe several types of policies [28]. Compared to administration APIs, the main benefits of using a DSL are to facilitate the non-intuitive and error-prone task of policy specification, to come with transformation mechanisms for simple integration of policies into the administration framework, and to allow coupling with verification tools for automated proof of security properties such as policy correctness or consistency. DSLs are thus interesting candidates to specify security adaptation policies, in turn guiding the selection of security functions to be applied to the system.

This paper tackles the issue of security adaptation strategy specification by proposing a *DSL for self-protection*. The DSL is based on the condition-action approach, using the notion of *GAP (General Adaptation Policy)* coupled with taxonomies of attacks and countermeasures to describe the mapping between reactions and intrusions during the system life-cycle. It also captures trade-offs between security and other concerns such as energy efficiency. A translation mechanism enabling to refine the DSL into a run-time representation, and to integrate the adaptation policy within a self-protection framework is presented. An example also illustrates how trade-offs between non-functional dimensions may guide the choice of the applied reaction.

The rest of this paper is organized as follows. After reviewing related work (Section 2), we provide some background on our approach to self-protection and on DSLs for autonomic systems (Section 3). We then present our DSL for specifying autonomic security management strategies (Section 4). We also show how it may be refined into an executable representation (Section 5). Finally, we conclude by describing ongoing and future research (Section 6).

2 Related Work

Self-protection frameworks apply the autonomic vision to the security domain [8]. Frameworks like Jade [9] or the self-defending platforms of Intel [2] enable reconfiguration of different protection functionalities based on evolution of the security context, but generally without addressing the specification of the autonomic security management strategy.

In more generic policy-based management frameworks [3,26], system execution is governed by applying predefined policies to respond to context changes – adaptation strategy specification was for instance discussed in [17,29]. Unfortunately, these frameworks hardly considered security, with the notable exception of [26]. The specification and implementation of adaptation policies taking into account different concerns, such as trade-offs between security and other dimensions, remains an open issue.

Advances in security context modeling also provide useful inputs for security adaptation decisions, abstracting security-related data into a formal model [1]. Ontologies are particularly helpful to structure or characterize security information [18], for instance to classify attacks [24], intrusions [27] or security context [10]. They allow better communication and reuse of the defined concepts, are

extensible, and also enable to reason about some security enforcements. Unfortunately, because of the diversity of targeted domains, a unified security ontology, generic but still able to describe in depth and reason about practical aspects of security still remains to be found [6].

DSLs are another modeling approach based on specification languages addressing a particular domain. A DSL offers primitives to describe a given domain and some implementation mechanisms such as annotations for integration into a run-time [28]. Platforms like Tune [7] and Kermeta [21] provide tools to generate various DSLs and their corresponding IDEs. However, a DSL for describing security adaptations seems to be missing, along with language mechanisms for integration into autonomic frameworks.

Like our approach, the ReD (Reaction after Detection) framework enables to choose new authorization policies to counteract network intrusions [13,12]. Based on OrBAC ontologies, ReD may support multiple types of authorization policies (e.g., integrating information flow requirements [5]) by mapping OrBAC to a security model through the definition of the corresponding ontology [10].

3 Background

We now provide some background on our self-protection framework (Section 3.1), explaining the benefits of DSLs for autonomic security strategy specification (Section 3.2), and describing the different actors involved (Section 3.3).

3.1 The ASPF Framework

To realize context-aware autonomic adaptations, the policy-driven paradigm has successfully demonstrated its power and generality [25]: system functionalities are governed by a set of policies and, as the context changes, new policies may be selected to activate in the system functions better adapted to its new environment. The aim of *ASPF (Autonomic Security Policy Framework)* [15] was to apply that approach to self-managed security, by describing the design of an autonomic security manager for pervasive systems.

A pervasive network is modelled as organized into several *clusters*, each containing a set of *nodes*. Each cluster is managed by a *cluster authority*, which controls the cluster structure – nodes may join or leave the cluster dynamically – and enforces a *cluster authorization policy* which is applicable to all nodes in the cluster. Nodes have various resource limitations, ranging from sensors to laptops, and enforce different *node authorization policies*.

The overall self-protection framework [14] is divided into 3 abstract layers (see Figure 1). For each node, an *execution space* provides a run-time environment for application- or system-level resources, represented and manipulated as components. Node security management is performed in the *control plane* through the *VSK (Virtual Security Kernel)* OS architecture [16] which controls the execution space, both for application-specific customizations and enforcement of authorizations to access node resources. Finally, a distributed *autonomic plane*

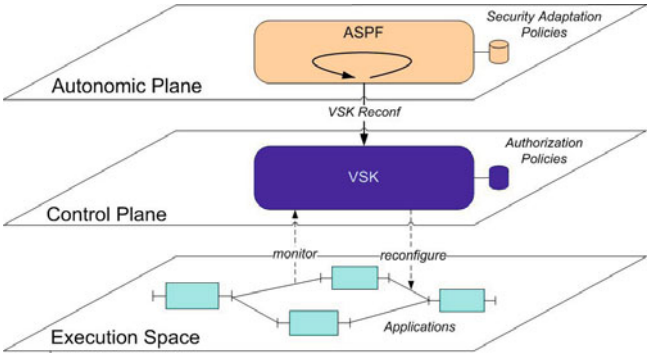


Fig. 1. A 3-Level Architecture for Self-Protection

supervises the authorization policies at cluster and node levels, and performs the necessary adaptations using several feedback loops. The general idea is to adapt security functionalities based on evolution of the security context, for instance by tuning the strength of authorization policies to the ambient risk level. The ASPF framework describes the organization of this last layer. However, although the different components of the feedback loops are well identified, the specification of the adaptation strategy which governs the behavior of the autonomic plane is still an unsolved issue.

3.2 The DSL Approach

A *Domain-Specific Language (DSL)* may be defined as "*a language tailored to a specific application domain*" [20]. DSLs present many benefits in terms of expressivity and simplicity of use, which led [7, 21] to specifying adaptation policies using DSLs in autonomous systems. To tackle the definition of the security adaptation strategy, we follow the same approach, but applying it to the security domain, resulting in a *DSL for self-protection*.

Several elements led us to express such strategies with a DSL, instead for instance of a simple management API. A self-protection framework should be able to react to security-relevant events occurring at run-time, either in the system itself or in its environment. Specifying the corresponding adaptation policies is a non-intuitive and error-prone task which may be facilitated with a security-oriented DSL. The DSL may serve to specify adaptation policies. It may also come with some transformation mechanisms for easy integration of the policies into the self-protection framework.

Moreover, in our current self-protection framework, adaptation strategies are purely action-based. However, higher-level strategies using objective or utility function policies are also desirable [17]. By enabling the specification of governance strategies with richer types of policies, the DSL approach should allow

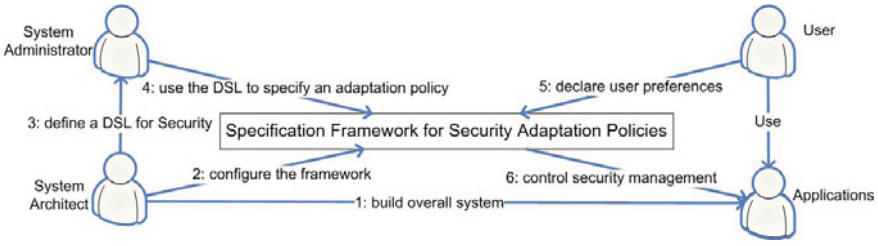


Fig. 2. Definition and Use of Security Adaptation Policies

describing self-managed security at different levels of granularity which can be refined (e.g., with notions of policy continuum [23]), and thus evolve towards greater autonomic maturity in the corresponding systems.

3.3 Main Actors of Autonomic Security Management

Security adaptation policies should be viewed as high-level protection strategies that affect and guide execution, usually expressed as rules defining reactions to specific security events. Several stakeholders with clearly-separated roles may cooperate to define and use such policies (see Figure 2):

- The *system architect* is the designer of the overall system. He implements a set of applications, configures the underlying protection framework, and defines the adaptation policy language with which the system administrator may specify security adaptation policies.
- The *system administrator* is in charge of the execution phase of the system. He has general knowledge about the system architecture and state, and specifies adaptation policies to guide the behavior of the protection framework.
- *Applications* are software entities the security of which is regulated by the self-protection framework.
- *Users* specify their preferences about adaptations of the system behavior from a usability perspective.

Security adaptation policies thus plays a central role in autonomic security management since they: (1) are described in a language (the DSL) designed by the system architect; (2) are specified by the system administrator; (3) may be customized by the user; and (4) guide the behavior of the self-protection framework. The DSL design should thus meet requirements from all stakeholders.

4 A DSL for Self-protection

We now describe the DSL itself which is based on the notion of *Generic Adaptation Policy (GAP)*. GAPs may be applied to different concerns such as security or energy efficiency, with possible trade-offs between concerns [4]. In the security case, GAPs are complemented with two taxonomies of attacks and countermeasures to specify security adaptation policies.

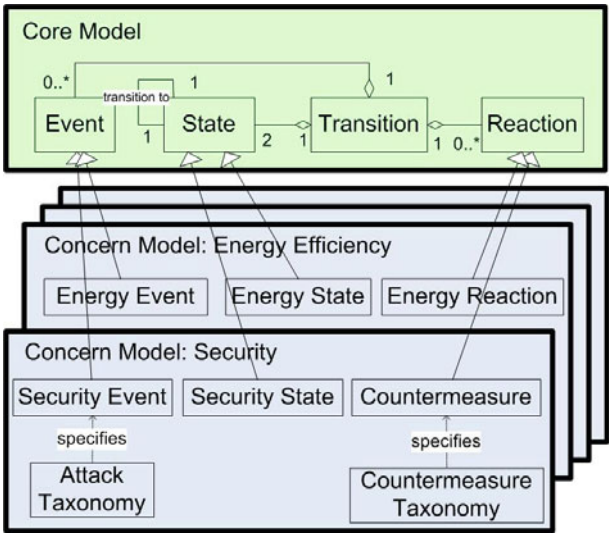


Fig. 3. DSL Meta-Model

As shown in the meta-model of Figure 3, the main concepts of our DSL are the following: the *State* class captures the status of the running system viewed from different concerns (risk level, energy efficiency, QoS, etc.); the *Event* class describes external signals which may trigger adaptations (this class may be specialized for each concern); the *Transition* class indicates the start and target states in an adaptation; and the operations to perform – captured by the *Reaction* class.

For each concern, different *concern models* abstract system features from the concern perspective by refining those classes. As our adaptation policies for self-protection specifically address the security concern, events and reactions are respectively specified by the *attack* and *countermeasure taxonomies* (described next). However, other concerns like energy efficiency may also be integrated into the adaptation policy using specific taxonomies.

4.1 Taxonomy of Attacks

This taxonomy classifies potential attacks or security-relevant events calling for a modification of the system protection settings. A sample attack taxonomy is shown in Figure 4, but others may also be used as well [24], the DSL not being tied to a specific taxonomy or ontology. Attacks may be classified according to confidentiality, integrity, and availability (CIA) security objectives. In turn, each class of attacks contains a set of specific potential attacks that may occur in the system. For instance, the *Packet Flooding* attack that disturbs the network by sending a large number of packets may be part of the *DoS* attack class threatening availability.

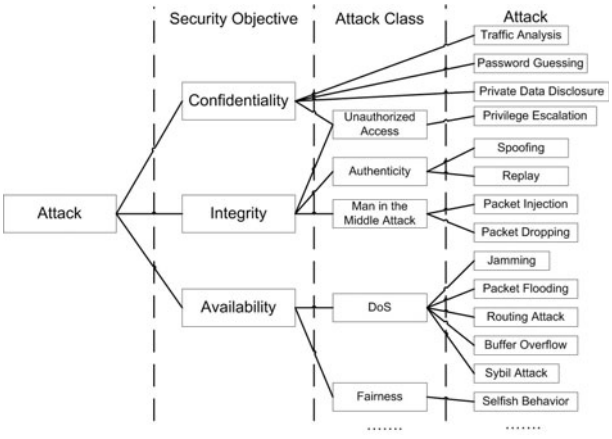


Fig. 4. Attack Taxonomy

To specify the adaptation policy, the system administrator identifies the most relevant attacks in the taxonomy. The corresponding security events may then be used in the GAP to trigger the adequate reactions.

4.2 Taxonomy of Countermeasures

The countermeasure taxonomy classifies the reactions applicable to respond to security-sensitive events. A sample taxonomy is shown in Figure 5. As for the attack taxonomy, security functions are structured according to CIA security objectives. For example, the *Strengthen Encryption* countermeasure class is classified as a reaction improving confidentiality.

To specify the adaptation policy, based on the identified attack and security objective to guarantee, the system administrator will select from the taxonomy the most appropriate countermeasure to trigger – a more automated approach is for instance described in [11].

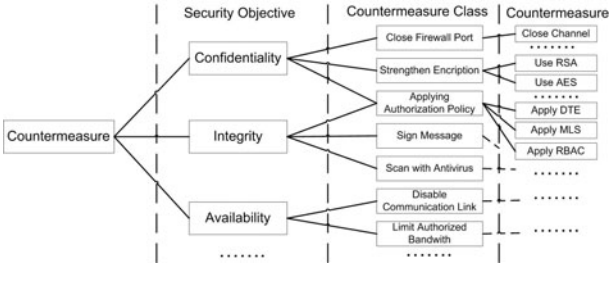


Fig. 5. Countermeasure Taxonomy

4.3 Generic Adaptation Policies (GAP)

Security adaptation policies should be intuitive for user-friendly specification, which is not the case of hard-coded if-then-else rules. We thus prefer to use event-based policies, instantiating the meta-model presented previously to specify states, events, transitions, and reactions to describe possible adaptations. The result is captured by the notion of GAP, formally defined as a tuple $[V, E, AT, gap]$ where:

- $V \subseteq V_1 ... \times V_i \times ... \times V_m$ is the *state space* of the running system. Each V_i represents a system state dimension identified by the system architect, e.g., energy efficiency, risk level, etc. It may be seen as a state variable, upon the values of which system adaptations will be performed. The system state is then given by a tuple $[v_1, ..., v_i, ..., v_m]$ of that space, where $v_i \in V_i$ is the value of each state variable.
- E is a finite set of *events* e that may occur in the system according to system evolution or context change. Events e are mainly related to monitoring infrastructures like intrusion detection systems, and will trigger adaptations. The events are partially derived from the attacks defined in the attack taxonomy.
- AT is a finite set of *adaptation reactions at* that can be applied to the running system. Different types of reactions may be performed, and are described by the system administrator. In the context of self-protection, the reactions are derived from the countermeasure taxonomy.
- The $gap : V \times E \rightarrow 2^{V \times AT}$ function maps a current state v and received event e to a set $gap(v, e)$ of proposed reactions and foreseen destination states.

Specifying a GAP then amounts to describing the transitions in terms of adaptation reactions and destination states from a set of initial states and events. Such a specification may be compactly represented using transition diagrams, as shown in the following example.

4.4 An Example of GAP Specification

We consider a system with only one state dimension, the risk level, which represents the vulnerability of the system, classified in the following 4 levels: $V = V_1 = \{\texttt{very hostile}, \texttt{hostile}, \texttt{neutral}, \texttt{friendly}\}$.

Two attacks are considered from the attack taxonomy, *Packet Flooding* and *Privilege Escalation*: $E = \{e_0, e_1\}$ (see Table 1). The *Packet Flooding* attack belongs to the *DoS* class compromising availability. The *Privilege Escalation* attack is part of the *Unauthorized Access* attack class weakening confidentiality and integrity.

We consider three reactions from the countermeasure taxonomy, *Close Channel* (r_0) for the availability security objective, and *Apply Domain Type Enforcement (DTE) authorization policy* (r_1) and *Use RSA* (r_2) for the confidentiality and integrity objectives: $AT = \{r_0, r_1, r_2\}$ (see Table 2). The table also indicates a subjective assessment of the protection strength of each countermeasure, to give an idea of its effectiveness to decrease the risk level.

Table 1. GAP Event Specification (Attack Taxonomy)

Event ID	Event Name	Event Type
e_0	Packet Flooding	DoS
e_1	Privilege Escalation	Unauthorized Access

Table 2. GAP Reaction Specification (Countermeasure Taxonomy)

Reaction ID	Reaction Name	Reaction Type	Protection Effect
r_0	Close Channel	Disable Communication Link	+++
r_1	Apply DTE Policy	Apply Authorization Policy	++
r_2	Use RSA	Strengthen Encryption	+

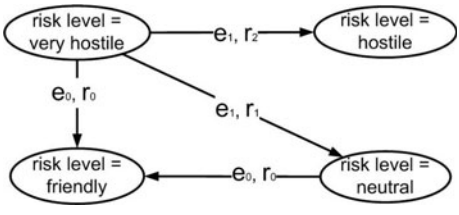


Fig. 6. A GAP Sample Specification

The system administrator then may define the *gap* function using the transition diagram shown in Figure 6. For instance, transition (e_1, r_2) from state ($risk =$ `very hostile`) to state ($risk =$ `hostile`) means that if a *Privilege Escalation* attack is detected in an already very hostile environment, encrypting communications may help decrease somewhat system vulnerability.

5 From DSL to Run-Time Representation

In this section, we show how the adaptation policies may be converted into high-level autonomic guidelines executable within a self-protection framework.

GAPs describe adaptation policies using events, states, etc. But from an implementation perspective, GAP policies are neither dependable nor efficient for run-time control since: (1) for each situation, more than one adaptation path may be proposed, making selection difficult and time-consuming; and (2) specified policies could be incomplete by construction, automated policy checking being complex and CPU-intensive due to the dimension of the state space. A more compact representation of adaptation policies called *Autonomic Adaptation Policies (AAP)* is thus defined to improve both dependability and efficiency. Some mechanisms are also proposed to translate GAPs to AAPs.

As shown in Figure 7, the main steps of the DSL life-cycle are the following:

1. A security expert of intrusions defines the attack taxonomy.
2. A security expert of reactions defines the countermeasure taxonomy.

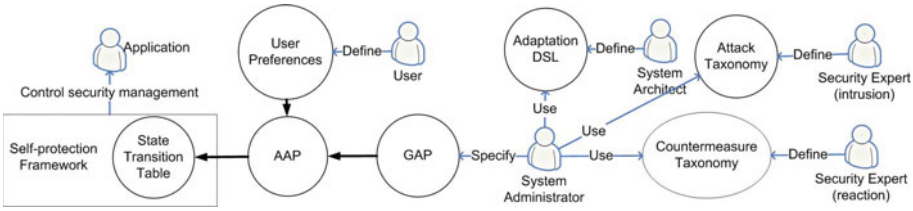


Fig. 7. Adaptation Policy Transformation

3. The system architect defines the syntax and semantics of the DSL in specifying the set of states V, the set of events E and the set of reactions AT.
4. The system administrator specifies the set of transitions of the GAP based on the previous sets of states, events, and reactions.
5. The user defines his preferences for security adaptations, e.g., prefer security over energy efficiency.
6. The translation mechanisms integrate the user preferences into the GAP, generate an AAP, and transform the AAP to a state transition table as run-time representation of the security adaptation policy.

In this process, each actor works in his own domain, translation mechanisms weaving together their different inputs into a run-time representation. Roles are clearly separated, policy specification and checking becoming independent tasks.

5.1 Autonomic Adaptation Policies (AAP)

For proper autonomic decision-making, a run-time adaptation policy should be both *complete* and *deterministic*. For any execution situation, at least one adaptation should be proposed to maintain execution continuity. A GAP should thus satisfy:

Property 1 (Completeness). *A GAP is complete if it leaves no situation unspecified, so that at least one adaptation reaction is associated to each event-state pair, i.e.,* $\forall(v,e) \in V \times E, |gap(v,e)| \geq 1$.

Besides, to avoid the ambiguity of adaptation decisions, different adaptations for the same conditions should not be possible. This property makes the security adaptation strategy highly sensitive information: knowledge how the system will react upon attacks could be used by intruders to defeat counter-measures more easily. We thus assume security adaptation strategies to be well protected. A GAP should thus satisfy:

Property 2 (Determinism). *A GAP is deterministic if no more than one adaptation may be realized in a given situation, so that at most one reaction is proposed for each event-state pair, i.e.,* $\forall(v,e) \in V \times E, |gap(v,e)| \leq 1$.

We define an AAP as a GAP fulfilling the two previous properties. The separation of roles of the different stakeholders is thus effectively achieved since the

system administrator only has to define the GAP specification, while property verification and GAP-AAP translation will be performed by DSL compile-time and run-time mechanisms.

5.2 A Sample Translation

The following example illustrates how the translation from GAP to AAP may be performed.

GAP Definition. We consider a pervasive system where security adaptation policies $GAP = (V, E, AT, gap)$ are defined as follows:

- $V \subseteq V_1 \times V_2$ represents the state of the system with 2 dimensions: the risk level (V_1) and the energy consumption (V_2). The *risk level* captures the vulnerability of the whole system, classified into 4 levels. The *energy consumption* is related to the device energy efficiency, with also 4 levels. Thus $V_1 = \{\texttt{very hostile}, \texttt{hostile}, \texttt{neutral}, \texttt{friendly}\}$, and $V_2 = \{\texttt{critical}, \texttt{high}, \texttt{normal}, \texttt{low}\}$. We consider only 5 states as detailed in Figure 8.
- E is the set of alarms raised by an intrusion detection system. To simplify, we only consider two events: $e_{\texttt{attackDetected}}$ when an attack is detected, and $e_{\texttt{returnSafe}}$ indicating the return to a safe state after application of a countermeasure. Then $E = \{e_{\texttt{attackDetected}}, e_{\texttt{returnSafe}}\}$.
- The reactions AT are defined as applying authorization policies into the system. Four policies p_1, p_2, p_3, p_4 are used as countermeasures. In using $\texttt{Permission}(p_i)$ to represent the set of all permissions of the policy p_i for a fixed set of subjects and objects, we define $p_i \succ p_j$ in terms of security strength which means that $\texttt{Permission}(p_i) \subset \texttt{Permission}(p_j)$. We then make the assumption that the policies are ordered as $p_4 \succ p_3 \succ p_2 \succ p_1$ in terms of the security strength which means that $\texttt{Permission}(p_4) \subset \texttt{Permission}(p_3) \subset \texttt{Permission}(p_2) \subset \texttt{Permission}(p_1)$. We also assume that the policies are ordered as $p_2 \succ p_4 \succ p_3 \succ p_1$ in terms of the energy consumption.
- *gap* specifies the allowed adaptations, captured as transitions between states based on received events. Adaptation rules include the initial state, upcoming events, actions to perform, and destination state for each adaptation. Possible evolutions of the system may be represented by a transition diagram as shown in Figure 8.

For instance, the transition from state s_2 ($risk = \texttt{neutral}, energy = \texttt{high}$) to state s_2 ($risk = \texttt{friendly}, energy = \texttt{critical}$) on event $e_{attackDetected}$, with reaction p_4 means that if the risk level is `moderate`, the energy consumption already `high`, and that an attack is detected, the authorization policy p_4 will be applied, driving the system in a state where the risk is decreased to `friendly`, but the energy consumption being increased to `critical`.

GAP → AAP Translation. Note that there are no adaptations associated to the state s_4 ($\texttt{risk} = \texttt{neutral}, \texttt{energy} = \texttt{normal}$) as shown in Figure 8,

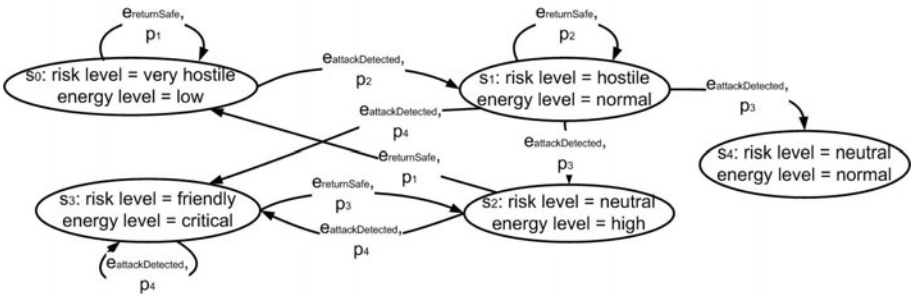


Fig. 8. A Typical Generic Adaptation Policy

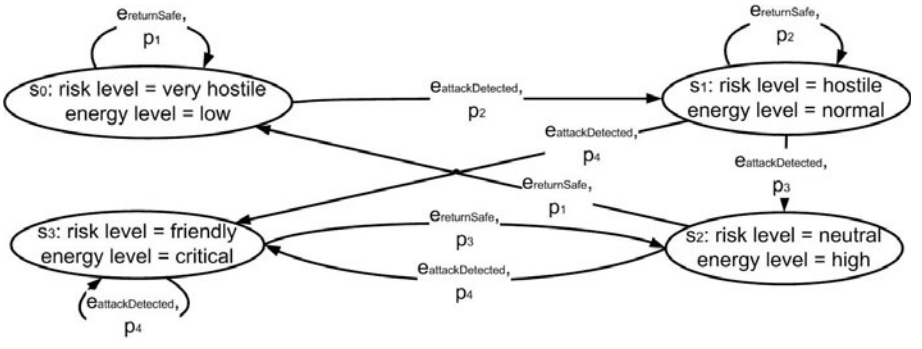


Fig. 9. GAP fulfilling Property 1

i.e., system execution will be blocked upon entering this state. Such states and the corresponding transitions should be eliminated from the GAP in order for Property 1 to hold. The system is then thus described only by the 4 states shown in Figure 9, at least one adaptation being associated to each state-event pair.

The specified policy may be initially non-deterministic, as for same state-event pair several adaptations may be proposed bringing the system to different destination states. For instance, in state s_1, on event $e_{attackDetected}$, the system may apply either p_3 or p_4 policies, respectively driving the system to states s_2 and s_3. This makes adaptation decisions difficult.

To retrieve a deterministic policy, we use a utility function $uFun$ to assess the utility of each destination state, and trim the states with low utilities. A utility value is thus attached to each state based on the user preferences. In the example, sample utilities of the 4 considered states are $uFun(s_0) = 0.1$, $uFun(s_1) = 0.2$, $uFun(s_2) = 0.4$, $uFun(s_3) = 0.3$. Only transitions with the highest utility values are kept, yielding the policy shown in Figure 10, which now also satisfies the determinism property. In the case where several transitions lead to the same highest utility value, one transition will be randomly selected.

The resulting GAP satisfies the two properties defined in the previous section, thus is as an AAP (see Figure 10). A *state transition table*, directly executable

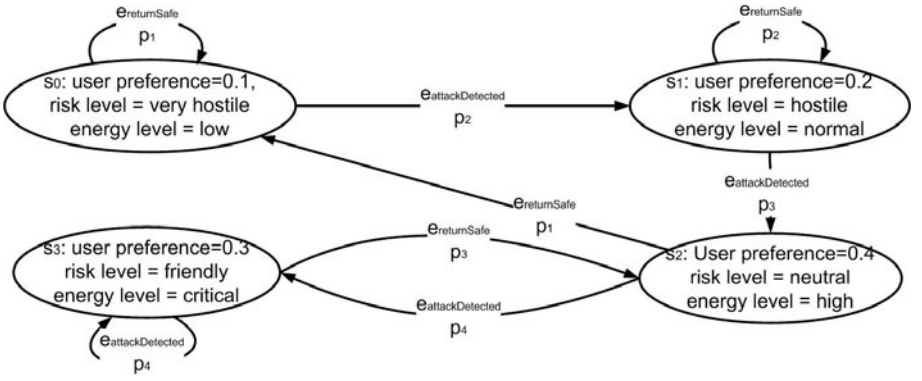


Fig. 10. GAP fulfilling Properties 1 and 2

in a self-protection framework, can be derived from this AAP. This table has been integrated in our self-protection framework in order to propose run-time unique adaptation reactions.

6 Conclusion

This paper presented a DSL to describe security adaptation policies in a self-protection framework. The DSL is based on the condition-action approach, and on a taxonomy of threats and applicable reactions. Different actors are separated in the policy specification process: the system administrator uses GAPs to specify policies intuitively. GAPs are then checked to guarantee run-time dependability of the policies, and translated into AAPs, suitable for integration at run-time into legacy self-protection frameworks. Trade-offs between security and other concerns may notably guide the refinement process.

Currently, a DSL framework called yTune is under development within the SelfXL project, including an editor and a parser for specification and checking of different DSLs. yTune should be seen as a meta-language for DSL definition and implementation. Ongoing work is focused around implementing the described refinement mechanisms for the self-protection DSL in the yTune parser, and coupling the DSL toolchain with the ASPF self-protection framework. In the future, we also plan to enhance the DSL with more complete and realistic taxonomies for security attacks and countermeasures, for instance through dedicated security ontologies which may be coupled with the corresponding security components to detect intrusions and perform reactions.

References

1. Workshop on Logical Foundations of an Adaptive Security Infrastructure (WOLFASI). In: conjunction with Workshop on Foundations on Computer Security, FCS (2004)

2. Agosta, J., et al.: Towards Autonomic Enterprise Security: Self-Defending Platforms, Distributed Detection, and Adaptive Feedback. Intel. Technology Journal 10(4) (2006)
3. Agrawal, D., Lee, K.-W., Lobo, J.: Policy-Based Management of Networked Computing Systems. IEEE Communications Magazine 43(10), 69–75 (2005)
4. Alia, M., Lacoste, M., He, R., Eliassen, F.: Putting Together QoS and Security in Autonomic Pervasive Systems. In: International Symposium on QoS and Security for Wireless and Mobile Networks (Q2SWinet) (2010)
5. Ayed, S., Cuppens-Boulahia, N., Cuppens, F.: An Integrated Model for Access Control and Information Flow Requirements. In: Cervesato, I. (ed.) ASIAN 2007. LNCS, vol. 4846, pp. 111–125. Springer, Heidelberg (2007)
6. Blanco, C., Lasheras, J., Valencia-Garcia, R., Fernandez-Medina, E., Alvarez, J., Piattini, M.: A Systematic Review and Comparison of Security Ontologies. In: International Conference on Availability, Reliability and Security (ARES) (2008)
7. Chebaro, O., Broto, L., Bahsoun, J.-P., Hagimont, D.: Self-TUNe-ing of a J2EE Clustered Application. In: International Workshop on Engineering of Autonomic and Autonomous Systems (EASe) (2009)
8. Chess, D., Palmer, C., White, S.: Security in an Autonomic Computing Environment. IBM Systems Journal 42(1), 107–118 (2003)
9. Claudel, B., De Palma, N., Lachaize, R., Hagimont, D.: Self-protection for Distributed Component-Based Applications. In: Datta, A.K., Gradinariu, M. (eds.) SSS 2006. LNCS, vol. 4280, pp. 184–198. Springer, Heidelberg (2006)
10. Coma, C., Cuppens-Boulahia, N., Cuppens, F., Cavalli, A.R.: Context Ontology for Secure Interoperability. In: International Conference on Availability, Reliability and Security (ARES) (2008)
11. Cuppens, F., Gombault, S., Sans, T.: Selecting Appropriate Counter-Measures in an Intrusion Detection Framework. In: IEEE Computer Security Foundations Workshop (CSFW) (2004)
12. Cuppens, N., Cuppens, F., Lopez de Vergara, J., Guerra, J., Debar, H., Vazquez, E.: An Ontology-based Approach to React to Network Attacks. In: International Conference on Risk and Security of Internet and Systems (CRiSIS) (2008)
13. Debar, H., Thomas, Y., Boulahia-Cuppens, N., Cuppens, F.: Using Contextual Security Policies for Threat Response. In: Büschkes, R., Laskov, P. (eds.) DIMVA 2006. LNCS, vol. 4064, pp. 109–128. Springer, Heidelberg (2006)
14. He, R., Lacoste, M.: Applying Component-Based Design to Self-Protection of Ubiquitous Systems. In: 3rd ACM Workshop on Software Engineering for Pervasive Services (SEPS) (2008)
15. He, R., Lacoste, M., Leneutre, J.: A Policy Management Framework for Self-Protection of Pervasive Systems. In: International Conference on Autonomic and Autonomous Systems (ICAS) (2010)
16. He, R., Lacoste, M., Leneutre, J.: Virtual Security Kernel: A Component-Based OS Architecture for Self-Protection. In: IEEE International Symposium on Trust, Security and Privacy for Emerging Applications (TSP) (2010)
17. Kephart, J., Walsh, W.: An Artificial Intelligence Perspective on Autonomic Computing Policies. In: IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY) (2004)
18. Kim, A., Luo, J., Kang, M.: Security Ontology for Annotating Resources. In: International Conference on Ontologies, Databases, and Application of Semantics, ODBASE (2005)

19. Lacoste, M., Jarboui, T., He, R.: A Component-Based Policy-Neutral Architecture for Kernel-Level Access Control. Annals of Telecommunications 64(1-2), 121–146 (2009)
20. Mernik, M., Heering, J., Sloane, A.: When and How to Develop Domain-Specific Languages. ACM Computing Surveys 37(4), 316–344 (2005)
21. Muller, P.-A., Fleurey, F., Jézéquel, J.-M.: Weaving Executability into Object-Oriented Meta-languages. In: Briand, L.C., Williams, C. (eds.) MoDELS 2005. LNCS, vol. 3713, pp. 264–278. Springer, Heidelberg (2005)
22. NIST. A Survey of Access Control Models. In: NIST Privilege (Access) Management Workshop (2009), http://csrc.nist.gov/news_events/privilege-management-workshop/
23. Serrano, M., van der Meer, S., Strassner, J., Paoli, S., Kerr, A., Storni, C.: Trust and Reputation Policy-Based Mechanisms for Self-protection in Autonomic Communications. In: González Nieto, J., Reif, W., Wang, G., Indulska, J. (eds.) ATC 2009. LNCS, vol. 5586, pp. 249–267. Springer, Heidelberg (2009)
24. Simmonds, A., Sandilands, P., van Ekert, L.: An Ontology for Network Security Attacks. In: Manandhar, S., Austin, J., Desai, U., Oyanagi, Y., Talukder, A.K. (eds.) AACC 2004. LNCS, vol. 3285, pp. 317–323. Springer, Heidelberg (2004)
25. Strassner, J., de Souza, J.N., Raymer, D., Samudrala, S., Davy, S., Barrett, K.: The Design of a New Policy Model to Support Ontology-Driven Reasoning for Autonomic Networking. In: Latin American Network Operations and Management Symposium (LANOMS) (2007)
26. Twidle, K., Dulay, N., Lupu, E., Sloman, M.: Ponder2: A Policy System for Autonomous Pervasive Environments. In: International Conference on Autonomic and Autonomous Systems (ICAS) (2009)
27. Undercoffer, J., Joshi, A., Pinkston, J.: Modeling Computer Attacks: An Ontology for Intrusion Detection. In: Vigna, G., Krügel, C., Jonsson, E. (eds.) RAID 2003. LNCS, vol. 2820, pp. 113–135. Springer, Heidelberg (2003)
28. van Deursen, A., Klint, P., Visser, J.: Domain-Specific Languages: An Annotated Bibliography. ACM SIGPLAN Notices 35(6), 26–36 (2000)
29. Verma, D., Calo, S.B., Cirincione, G.: A State Transition Model for Policy Specification. IBM Research Report RC24766 (March 2009)

Secure and Scalable RFID Authentication Protocol

Albert Fernàndez-Mir, Jordi Castellà-Roca, and Alexandre Viejo

Dpt. d'Enginyeria Informàtica i Matemàtiques, UNESCO Chair in Data Privacy, Universitat Rovira i Virgili, Av. Països Catalans 26, E-43007 Tarragona, Spain
{albert.fernandez,jordi.castella,alexandre.viejo}@urv.cat

Abstract. The radio frequency identification (RFID) enables identifying an object remotely via radio waves. This feature has been used in a huge number of applications, reducing dramatically the costs in some production processes. Nonetheless, it also poses serious privacy and security risks to them. Thus, researchers have presented secure schemes that prevent attackers from misusing the information which is managed in those environments. These schemes are designed to be very efficient at the client-side, due to the limited resources of the tags. However, they should be efficient at the server-side also, because the server manages a high number of tags, i.e. any proposal must be scalable in the number of tags. The most efficient schemes are based on client-server synchronization. The answer of the tag is previously known by the server. These kind of schemes commonly suffers desynchronization attacks. We present a novel scheme with two main features: (i) it improves the scalability at the sever-side; and (ii) the level of resistance to desynchronization attacks can be configured.

Keywords: rfid, identification, security, privacy, scalability.

1 Introduction

RFID (Radio Frequency IDentification) is a cutting-edge technology that uses radio waves in order to identify remote objects. A basic RFID scheme includes one reader and several small devices which are named *tags*. Each tag is embedded in the object which must be identified. The tag uses the energy of the wave sent by the reader to transmit its unique serial number to the reader. In this way, the reader identifies a certain item.

Technically, any item is capable of incorporating an RFID tag. Nevertheless, it should be done only in the cases where the relationship between the price of the tag and the price of the item is reasonably sufficient. The easy implantation of these devices paves the way to many novel applications of this technology. The use of this technology can reduce dramatically the costs in some production processes (*e.g.* the textile industry or the car industry among others).

Even though this technology offers important advantages to the users, it also poses serious privacy and security risks to them. According to that, it is necessary

J. Garcia-Alfaro et al. (Eds.): DPM 2010 and SETOP 2010, LNCS 6514, pp. 231–243, 2011.

to develop secure schemes that prevent attackers from misusing the information which is managed in those environments. This point is earnestly suggested by the European Union in [1].

The most vulnerable part of the system is the communication between the reader and the tag. During this transmission, an attacker can eavesdrop the messages (*e.g.* the identifier of the tag) and she can identify the item or monitor its position.

In the next section we analyze the different requirements that must be fulfilled by a secure RFID protocol. In Section 3, the current literature in this field is described. Section 4 presents our new proposal. Section 5 analyzes the security of the new protocol. Finally, some conclusions are given in Section 6.

2 Requirements for RFID Protocols

Before designing a new RFID identification protocol, it is important to take into account a number of requirements that must be achieved. These requirements are related to privacy, security and performance issues.

2.1 Privacy

Privacy is one of the main concerns in RFID systems. The use of radio waves to communicate the devices between them enable any attacker to eavesdrop the transmissions and obtain the identification of the tags. Song et al. [2] consider two different issues related to privacy:

- *Profile.* In a traditional RFID system, when the reader asks the tag, it responds with its unique identifier. An unauthorized reader that obtains a certain number of identifiers associated to a single user can create a user profile. Also, RFIDs which contain personal information (*e.g.* passport, medical card...) can disclose this sensible data in an unauthorized way.
- *Monitoring and location.* If the answers of a certain tag can be distinguished from the ones coming from other tags, an attacker can get its location using various readers located in strategic points.

2.2 Security

Below are the most common attacks which can be launched on RFID schemes:

- *Denial of service.* An attacker can jam messages (and eliminate) the messages which are exchanged between the different tags and the reader in order to disrupt the communications.
- *Tracking.* If a certain tag is always identified with the same identifier, an attacker can use various readers in order to know the approximate location of the tag and/or the person who is wearing it.
- *Forward security.* If an attacker is able to compromise a tag, then she can track that tag knowing its previous identifiers and knowing its location.

– *Impersonation of devices.* An attacker may impersonate a tag even without knowing its data. For example, in a replay attack, an attacker can listen to the identification of a certain tag and can forward the same message to the reader behaving like the original tag. Similarly, if an attacker knows the internal state of the tag, it is able to impersonate the reader. The combination of both situations results in a *man-in-the-middle* attack [3].

2.3 Efficiency

Due to the technological limitations of the RFID tags, designers must take into account the following requirements in order to develop an RFID identification protocol:

– *Minimizing the storage capacity.* The volume of data that is stored in the tag should be minimal due to the limited memory of these devices.
– *Minimizing the computational cost.* The computational cost required in the tag side should be minimal due to the lack of resources (*e.g.* energy, computational power) of the tags.
– *Minimizing the communication cost.* The volume of data that each tag can transmit per second is limited by the bandwidth which is available for RFID tags [4, 5].
– *Scalability.* The server must be able to identify a large number of different tags using the same radio station. The use of exhaustive search in its database can slow down the identification process.

3 State of the Art

There are different types of secure authentication protocols for RFID systems. This section describes two classes of protocols. The first one is based on asymmetric cryptography. This kind of cryptography provides more security to the

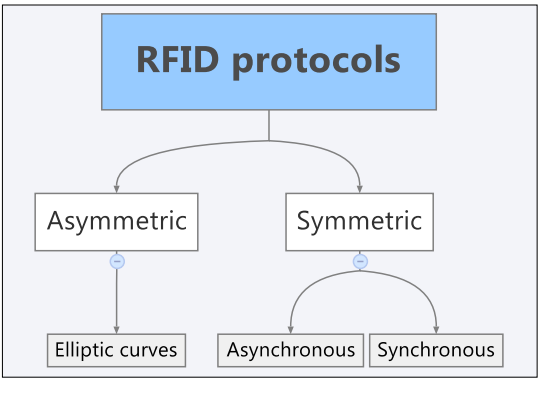


Fig. 1. Clasification of RFID authentication protocols

system. The second class is based on symmetric cryptography. Currently, this approach is the most significant and used because it can be implemented using a minor number of logic gates.

3.1 Asymmetric Protocols

The use of asymmetric cryptography is possible in RFID schemes. In this way, the work presented in [6] uses elliptic curves. Nevertheless, the limited resources of the tags limit the implementation of this kind of protocols: these schemes require a high number of logic gates to be executed and a high number of logic gates implies a higher cost for the tag. Therefore, in the present day their implementation is unfeasible.

3.2 Symmetric Protocols

Nowadays, this kind of protocols are the most used. Symmetric cryptography fits better in low-cost tags. Nevertheless, it is more difficult to achieve a good privacy level. These protocols can be divided into two categories: asynchronous and synchronous.

Asynchronous protocols. All the protocols found into this category base their security on one-way hash functions. From knowledge of the hash function h and $y = h(x)$, an intruder is not able to determine x. Hash-based protocols are very efficient in the tag side because the operations which are performed are minimal. According to that, they are really suitable for lightweight RFID schemes like the ones which are currently being used.

The first protocol based on hash functions was proposed by Rivest et al. [7] in 2003. In this work, the use of deterministic hash locks was presented. The OSK protocol [5] works in a similar way but here the IDs are updated after each identification using hash functions. This behaviour prevents intruders from tracing the IDs of the tags. This protocol also suffers from scalability issues since it is necessary to make a thorough search for each identifier until the right one is found. Avoine and Oechslin addressed this shortcoming in [8] that reduces the scalability problem but is vulnerable to replay attacks.

Juels [9], in 2006, proposed one of the most popular protocols in this field: the random hash locks. In this protocol, both the reader and the tag generate a random number r. The tag concatenates r with its unique ID and obtains a hash value $y = h(r||ID)$. The tag sends y to the reader and this device searches its database for the ID that matches y. Note that the reader has to compute $y_i = h(r||ID_i)$ for each ID_i in the database until the correct one is found. In the worst case, this process is done as many times as tags are controlled by the system. The main problem of this method is again the scalability of the system [10].

Despite the efforts made in this field, current protocols still have serious scalability problems at the server side when the number of tags is very high.

Synchronous protocols. This kind of protocols appear because of the scalability issues related to the hash-based protocols. Several synchronous protocols are also based on hash functions. Nevertheless, their main particularity is the use of state synchronism between the reader and the tag. In this way, if both devices are synchronized, these protocols enable quick and secure identification because the server knows the next tag response. Protocols like [11, 12] update the identifiers of the tags but still remain vulnerable to traceability attacks.

Regarding scalability, these protocols are scalable while both devices are synchronized but they have problems when the synchronism is lost. The YA-TRAP protocol [13] uses a series of time states $[T_0, ..., T_{MAX}]$ that protects the identity of the tag.

Dimitriou [14] in 2005 and Lee et al. [15] in 2006, proposed two new protocols based on synchronization between server and tag. Even so, their proposals had did not prevent that an attacker could track a tag. To solve this problem Ha et al. [16] use the synchronization state for fast identification. They use hash locks to identify the tags when they are desynchronized. This is a serious problem in terms of scalability because if an attacker disrupts the update messages, the system does not scale correctly.

The problems arise when an attacker desynchronize the two devices. Other problems in these types of protocols appear when an attacker tries to track a tag. However, several protocols based on synchronism [17–20], have different security issues that are described in [21] as for example: tracking, impersonation of devices or denial of service.

4 Our Proposal

As seen in Section 3, the protocols that are efficient at both server and client sides, are those based on synchronization. However, as detailed in [21] these protocols are vulnerable to desyncronization attacks or when they are attacked, they present scalability problems. We present a protocol that is efficient for both the client and the server in the synchronization state. Besides, if the protocol is not synchronized, the level of resistance against denial of service attacks can be selected and the protocol will still be robust against privacy threats. The proposed scheme has been designed considering that an attacker must be unable to distinguish whether a tag is synchronized or not. This fact adds one more degree of difficulty to an attacker who wants to desynchronize the system.

The proposed protocol consists of four phases: initialization phase, synchronized identification phase, desynchronized identification phase and update phase. The last one is executed after each successful identification. If the tag cannot be updated and the system is desynchronized, it can still be identified with the desynchronized identification phase preserving the initial security and privacy properties.

Table 1 shows the notation used to describe the new proposal.

Table 1. Notation used in the protocol

id	Tag identifier
R	Reader
T	Tag
r_i	Random number i
$h()$	Unidirectional Hash Function
$h_k()$	Keyed Hash Function (HMAC)
ks	Secret key of reader
$PRNG$	Pseudo-Random Number Generator
$SYNC$	Synchronization State
C_i	Pseudo-Random bit sequence
S	Pseudo-Random bit sequence
m_i	Update message i
k_δ	$h_{id}(r_1)$ done δ times
$\|\|$	Concatenation Operator
$\oplus$	XOR Operator

Table 2. BD values

$Table\,k$	$h'_{id}(r_1)$	r'_1	$h_{id}(r_1)$	r_1	$h_{ks}(id)$	id
r_1 r'_1						
k_1						
k_2						
k_3	...	...	...	...	...	...
...						
k_{MAX}						

4.1 System Environment

The proposed protocol requires a server that hosts its own database and a reader that transmits information from the various tags which are present in the scenario.

Each tag has a record in the server as shown in Table 2, and the server stores the following data for each tag:

- id: The identity of the tag.
- $h_{ks}(id)$: Hash value, where ks is a secret key that is only known by authorized readers and it used to generate the hash value.
- r_1 *and* r'_1: These random values and their associated hash values $h_{id}(r_1)$. Current random value (r_1) and next random value (r'_1) are kept in order to distinguish the state of synchronization of the system.
- $Table\,k$: This table is used to identify the tag in case of the system is not synchronized. We keep two columns of data corresponding to the values of the key r_1 previous and present used in secondary identification phase.

Another value that must be taken into account is MAX. This value is used to set the level of resistance to denial of service attacks. MAX is the is the length of the $Table\,k$ and the possible values that the server knows to identify a tag in a desynchronized identification phase.

4.2 Protocol Phases

In this section we describe the different phases of our protocol. Table 3 details the four phases and their steps.

Initialization phase. In this first phase, the server sends to the reader the following information: $h_{ks}(id)$, r_1 and $h_{id}(r1)$. This information will be used later by the tag to perform the authentication phase. The data is transmitted using a secure channel that guarantees their confidentiality.

Synchronized identification phase. When the tag and reader are synchronized, they run this phase to identify the tag. The system follows the next steps: the reader sends a random number r_0 to the tag. Then, the tag answers with the following information: $h_{id}(r_1)$, C_0 and r_2, where $C_0 = PRNG(r_1||r_0)$ and r_2 is a random value that is generated by the tag.

Upon reception of this data, the reader forwards it to the server. Then, the server computes r_1 and id. Besides, the server must be able to verify the tag sequence C_0 to ensure that it is authentic. This is done to avoid a phishing attack or a replay attack.

After sending all the data, the tag goes to a desynchronized state ($SYNC = 0$) until the update phase ends.

Update phase. This phase is used to update the values of the tag and avoid several attacks in order to protect the privacy. This phase is executed after synchronized an desynchronized phases.

First of all, the server composes m_0 concatenating a new $r_1^{'}$ and new $h_{id}(r_1^{'})$, these elements will be updated by the tag after the update phase. Then, the server generates m_1 that will be used by the tag to verify m_0. Later, the server concatenates m_0 and m_1 into m_2, generates the pseudorandom sequence S where $h_{ks}(id)||r_1$ is the seed and it finally computes $C_1 = m_2 \oplus S$. Finally, the server sends C_1 to the tag and the columns r_1 and r_1' of the $Table\,k$ are updated using the new key r_1'.

When the tag receives the message C_1, obtains the pseudorandom sequence $S^{'}$ with the same seed as the server $(h_{ks}(id)||r_1)$ and decrypts the message C_1 by a XOR operation. Then checks whether m_1 is correct or not. If it is correct, then the tag updates the new values $r_1^{'}$ and $h_{id}(r_1^{'})$ and sets $SYNC = 1$.

Desynchronized identification phase. This phase is executed when both devices are desynchronized. If the message C_1 corresponding to the update phase is incorrect or it has been received incorrectly (and assuming that $SYNC = 0$), the tag will follow the next steps: First of all the reader sends r_0 to the tag like

Table 3. Proposed authentication protocol

Server	**Reader**	**Tag**
Initialization phase (with secure channel)		
	$h_{ks}(id), r_1, h_{id}(r_1)$ →	
Synchronized identification phase		
	$r_0 \in R\{0,1\}^*$ r_0 →	$r_2 \in R\{0,1\}^*$ $C_0 = PRNG(r_1 \| r_0)$ $SYNC = 0$
$h_{id}(r_1), C_0, r_2, r_0$ ←	$h_{id}(r_1), C_0, r_2$ ←	
Search $h_{id}(r_1)$ Check C_0		
Update phase		
$m_0 = r_1^{'} \| h_{id}(r_1^{'})$ $\alpha = h_{ks}(id) \| r_2$ $m_1 = h_\alpha(m_0)$ $m_2 = m_0 \| m_1$ $PRNG(h_{ks}(id) \| r_1) = S$ $C_1 = m_2 \oplus S$		
C_1 →	C_1 →	$S^{'} = PRNG(h_{ks}(id) \| r_1)$ $m_3 = C_1 \oplus S^{'}$ $\alpha^{'} = h_{ks}(id) \| r_2$ $m_1 \stackrel{?}{=} h_{\alpha^{'}}(m_0)$ $r_1 = r_1^{'}, h_{id}(r_1) = h_{id}(r_1^{'})$ $SYNC = 1$
Desynchronized identification phase		
	$r_0 \in R\{0,1\}^*$ r_0 →	$r_2 \in \mathbb{R}\{0,1\}^*$ $\delta = \delta + 1$ $k_\delta = h_{r_1}(k_{\delta-1})$ $C_0 = PRNG(h_{r_1}(r_2 \| r_0))$
k_δ, C_0, r_2 ←	k_δ, C_0, r_2 ←	
If k_δ is not in BD Calculation of id by k_δ Check C_0 Start Update phase		

in the synchronized identification phase. Then the tag generates a new r_2 and, for $k_0 = h_{id}(r_1)$, performs the following computations:

$$\delta = \delta + 1$$

$$k_\delta = h_{r_1}(k_{\delta-1})$$

Next, the reader forwards the data to the server that will search for the value k_δ in the database like in the synchronized identification phase. If this value is not found, the server will search k_δ in the $Table\,k$ of each tag. If it finds the value, the identifier of the tag will be checked using C_0. If the server has two or

more records of k_δ in the database, it will obtain the correct identifier through checking the C_0 value. After identifying the tag, the reader starts the Update phase.

In case of desynchronization, the server has MAX opportunities to identify the tag before this becomes useless. The value of this parameter is discussed in the subsection 4.3.

4.3 Selection of Parameter MAX

The parameter MAX which is used in the protocol influences the system in the following three concepts:

- *Memory Space.* Considering n tags, in the BD there is a table of MAX values associated to each tag. Each of these values is equivalent to the output of a hash function. The output of the hash function SHA-1 [22] has a length of 160 bits. As a result, the memory requirements for this concept are $n \cdot MAX \cdot 160$ bits.
- *Computational cost.* In the secondary identification phase, k_δ must be found in the table of MAX positions which is associated to each tag. When k_δ is located, the reader knows the *id* of the corresponding tag. The tag associated to k_δ is located with a cost of $O(1)$ because it is only necessary to know whether this element is missing or not. As a result of that, the cost to locate the tag that contains k_δ in its table is O (1) because the server only need to know if this element is missing or not in our database. Therefore, the system scales correctly for large values of n.
- *Security.* The system can be configured to resist MAX attacks of desynchronization, *i.e.* a Denial of Service (DoS). This is justified in Section 5.

As a conclusion, the value of MAX depends on the available memory space and time β in which an attacker is able to get the MAX values of table k. For the calculation of this value, we consider an interval of 200 ms for each identification.

Table 4. Table of values MAX

MAX	*Tags*	*Disk space*	*Time* β
100	100	0,024 Gb	20 s
1.000	1.000	2,4 Gb	200 s
10.000	10.000	240 Gb	2000 s
100.000	100.000	24 Tb	5,5 h
1.000.000	1.000.000	2400 Tb	55,5 h

Although the memory requirements are high, we consider that this resource is cheap and therefore it is acceptable. In terms of time, it depends on the level of security which is needed by the system. Due to the current power of the processors, we believe that the user response time is not a major drawback for the system.

Table 4 presents different results depending on the MAX value in use and the number of tags in the system. Considering parameter β, it can be observed that for large values of MAX, the time which is necessary to obtain all the records in table k is high enough to avoid any attack. Nevertheless, the shortcoming of this security is the memory space which is needed.

5 Security Analysis

The protocol has the following properties of security and privacy:

- *Private identity.* A tag does not know its own *id*. It only knows $h_{ks}(id)$. Thus, an attacker who is able to get the contents of the tag will not know its true identifier.
- *Locating and tracking a certain tag.* Location privacy is guaranteed because, in each identification, the data sent is always different. In this way, values $h_{id}(r_1)$, C_0 and r_2 are updated in each identification. If the reader and the tag are not synchronized, then the updated elements are: k_δ, C_0 and r_2. If both devices are synchronized, $h_{id}(r_1)$ can appear twice with a probability of $1/2^{160}$. The probability for k_δ is the same as $h_{id}(r_1)$ in consecutive secondary identifications.
- *Impersonating a certain tag.* An attacker does not know values $h_{ks}(id)$, r_1, $h_{id}(r_1)$. These elements are sent in the initialization phase using a secure communication channel. An attacker trying to impersonate a certain tag without knowing these values will be detected by the reader.
- *Impersonating the server.* This case is similar to the last property. The server can only be impersonated by an attacker who knows the entire database. We assume that the database is hosted in a secure environment.
- *Replay attacks.* This type of attack is not possible in this protocol because in each identification, each one of the sides (the tag and the server) provides a new value which is computed at random. In this way, a replay attack at the server side will be successful only if the message which has been captured previously contains the expected r_0 value. The same happens at the tag side but with the value r_2. Random values r_0 and r_2 are generated in each device and they are unlikely to be repeated in a short period of time. In this way, value $h_{id}(r_1)$ is different in each Synchronized identification phase. In the Desynchronized identification phase case, k_δ is always different.
- *Forward security.* Our protocol is partially resistant to this kind of attacks. An attacker can obtain the following data: $h_{ks}(id), r_1, h_{id}(r_1)$. In order to link this data with the previous authentication (Synchronized identification phase), she should be able to compute r_1^* from the PRNG using sequence S,

where r_1' is the previous r_1^*. This operation is computationally hard since a secure PRNG is being used. Next, the attacker should be able to find the id used in the HMAC $h_{id}(r_1^*)$. Assuming that a secure HMAC function is being used, this attack is computationally unfeasible. However, the attacker could still identify the authentications done with the secondary method which have been computed using r_1.

– *Denial of service.* An attacker can query the tag or interrupt the update phase $MAX + 1$ times. So, the reader will not find k_δ in the database, *i.e.* the server cannot identify the tag. Since, MAX is a security parameter of our system, we can set this value in order to increase the resistance of the proposed protocol against this attack. If the table is large enough, the time which is needed to consume all the values will be high enough (this was discussed in Section 4.3).

Table 5, shows a comparison of security among several protocols of secure authentification on RFID technology.

Table 5. Comparison of security

Protocol	LCRP [14]	Juels et al. [9]	Lee et al. [15]	Ha et al. [16]	Our proposal
Information leakage	◯	◯	◯	◯	◯
Spoofing attack	◯	◯	◯	◯	◯
Replay attack	◯	◯	◯	◯	◯
Indistinguishability	×	◯	◯	◯	◯
Forward security	△	×	△	△	△
Resyncrhonization	×	◯	◯	◯	◯

◯: secure or support △: partially secure ×: insecure or not support

6 Conclusions and Future Work

We have presented a new protocol that is more scalable than the hash-based protocols or the synchronous proposals. We have used both technologies in order to achieve a cost of $O(1)$ for each identification. Besides, attackers are not allowed to know whether the system is synchronized or desynchronized.

As a secondary feature, tags do not know their own ids so an attacker who compromises a tag is not able to know its identifier.

Finally, the proposed scheme can be configured in order to resist a DoS attack. This is done by setting parameter MAX. For larger values of this parameter, the proposed system can stand better against DoS attacks but the memory space which is required is also higher.

In this way, the main shortcoming of the proposed scheme is the memory requirements. In scenarios with millions of tags, our database requires Terabytes of space. Fortunately, the market offer cheap solutions to address this point.

Future work will focus on implementing the proposed protocol in a real scenario. Current RFID low-cost tags have a very limited number of logic gates, hence they cannot be used to deploy our scheme. Fortunately, it is possible to use smart cards instead. This technology can offer a practical environment based on standards such as ISO 14443 and ISO 15693. In the future, low-cost tags with improved capabilities are expected to be available.

Disclaimer and Acknowledgments

The authors are solely responsible for the views expressed in this paper, which do not necessarily reflect the position of UNESCO nor commit that organization. This work was partly supported by the Spanish Ministry of Education through projects TSI2007-65406-C03-01 "E-AEGIS" and CONSOLIDER CSD2007-00004 "ARES", by the Spanish Ministry of Industry, Commerce and Tourism through project "eVerification" TSI-020100-2009-720 and by the Government of Catalonia under grant 2009 SGR 1135.

References

1. European Union, Commission recomendation of 12 May 2009 on the implementation of privacy and data protection principles in applications supported by radio-frequency identification. In Official Journal of the European Union (2009), http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2009:122:0047:0051:EN:PDF
2. Song, B., Mitchell, C.J.: RFID authentication protocol for low-cost tags. In: Proceedings of the First ACM Conference on Wireless Network Security, pp. 140–147 (2008)
3. Sarma, S., Weis, S., Engels, D.: RFID systems and security and privacy implications. In: Kaliski Jr., B.S., Koç, Ç.K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 1–19. Springer, Heidelberg (2003)
4. Avoine, G.: Cryptography in radio frequency identification and fair exchange protocols. In: Faculté Informatique et Communications, pp. 2007–06 (2005)
5. Ohkubo, M., Suzuki, K., Kinoshita, S., et al.: Cryptographic approach to privacy-friendly tags. In: RFID Privacy Workshop, vol. 82 (2003)
6. Martínez, S., Valls, M., Roig, C., Miret, J.M., Giné, F.: A secure Elliptic Curve-Based RFID Protocol. J. Comput. Sci. Tech. 24(2), 308–318 (2009)
7. Rivest, R., Weis, S., Sarma, S., Engels, D.: Security and privacy aspects of low-cost radio frequency identification systems. In: Hutter, D., Müller, G., Stephan, W., Ullmann, M. (eds.) PC 2003. LNCS, vol. 2802, pp. 454–469. Springer, Heidelberg (2003)
8. Avoine, G., Oechslin, P.: A scalable and provably secure hash-based RFID protocol. In: PERCOMW, pp. 110–114 (2005)
9. Juels, A.: Rfid security and privacy: A research survey. IEEE Journal on Selected Areas in Communications 24(2), 381–394 (2006)
10. Solanas, A., Domingo-Ferrer, J., Martínez-Ballesté, A., Daza, V.: A distributed architecture for scalable private RFID tag identification. Computer Networks 51(9), 2268–2279 (2007)

11. Kanf, J., Nyang, D.: RFID authentication protocol with strong resistance against traceability and denial of service attacks. In: Molva, R., Tsudik, G., Westhoff, D. (eds.) ESAS 2005. LNCS, vol. 3813, pp. 164–175. Springer, Heidelberg (2005)
12. Yang, J., Park, J., Lee, H., Ren, K., Kim, K.: Mutual authentication protocol for low-cost RFID. In: Handout of the Ecrypt Workshop on RFID and Lightweight Crypto (2005)
13. Tsudik, G.: YA-TRAP: Yet another trivial RFID authentication protocol. In: Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops, PerCom Workshops 2006 (2006)
14. Dimitriou, T.: A lightweight RFID protocol to protect against traceability and cloning attacks. In: Security and Privacy for Emerging Areas in Comunications Networks, SecureComm 2005, pp. 59–66 (2005)
15. Lee, S., Asano, T., Kim, K.: RFID: Mutual Authentication Scheme based on Synchronized Secret Information. In: Proceedings of the SCIS 2006 (2006)
16. Ha, J.C., Moon, S.J., Nieto, J., Boyd, C.: Low-cost and strong-security RFID authentication protocol. In: Denko, M.K., Shih, C.-s., Li, K.-C., Tsao, S.-L., Zeng, Q.-A., Park, S.H., Ko, Y.-B., Hung, S.-H., Park, J.-H. (eds.) EUC-WS 2007. LNCS, vol. 4809, pp. 795–807. Springer, Heidelberg (2007)
17. Lee, S., Asano, T., Kim, K.: RFID mutual authentication scheme based on synchronized secret information. In: Symposium on Cryptography and Information Security (2006)
18. Osaka, K., Takagi, T., Yamazaki, K., Takahashi, O.: An efficient and secure RFID security method with ownership transfer. In: Wang, Y., Cheung, Y.-m., Liu, H. (eds.) CIS 2006. LNCS (LNAI), vol. 4456, pp. 778–787. Springer, Heidelberg (2007)
19. Seo, Y., Lee, H., Kim, K.: A scalable and untraceable authentication protocol for RFID. In: Zhou, X., Sokolsky, O., Yan, L., Jung, E.-S., Shao, Z., Mu, Y., Lee, D.C., Kim, D.Y., Jeong, Y.-S., Xu, C.-Z. (eds.) EUC Workshops 2006. LNCS, vol. 4097, pp. 252–261. Springer, Heidelberg (2006)
20. Song, B., Mitchell, C.J.: A scalable and untraceable authentication protocol for RFID. In: RFID Authentication Protocol for Low-Cost Tags. In Wireless Network Security (WISEC), pp. 140–147 (2008)
21. Van Deursen, T., Radomirovic, S.: Attacks on RFID protocols. In: IACR eprint Archive 2008, vol. 310 (2008)
22. Eastlake, D., Jones, P.: US secure hash algorithm 1 (SHA1). In: RFC 3174 (2001)

Some Ideas on Virtualized System Security, and Monitors

Hedi Benzina* and Jean Goubault-Larrecq

LSV, ENS Cachan, CNRS, INRIA
61 avenue du Président Wilson, 94230 CACHAN, France
{benzina,goubault}@lsv.ens-cachan.fr

Abstract. Virtualized systems such as Xen, VirtualBox, VMWare or QEmu have been proposed to increase the level of security achievable on personal computers. On the other hand, such virtualized systems are now targets for attacks. We propose an intrusion detection architecture for virtualized systems, and discuss some of the security issues that arise. We argue that a weak spot of such systems is domain zero administration, which is left entirely under the administrator's responsibility, and is in particular vulnerable to trojans. To avert some of the risks, we propose to install a role-based access control model with possible role delegation, and to describe all undesired activity flows through simple temporal formulas. We show how the latter are compiled into Orchids rules, via a fragment of linear temporal logic, through a generalization of the so-called history variable mechanism.

1 Introduction

Intrusion detection and prevention systems (IDS, IPS) have been around for some time, but require some administration. Misuse intrusion detection systems require frequent updates of their attack base, while anomaly-based intrusion detection systems, and especially those based on statistical means, tend to have a high false positive rate. On the other hand, personal computers are meant to be used by a single individual, or by several individuals, none being specially competent about security, or even knowledgeable in computer science.

We propose to let the user run her usual environment inside the virtual machine of some virtualized architecture, such as Xen [28], VirtualBox [24], VMWare [25], or QEmu [15]. The idea has certainly been in the air for some time, but does not seem to have been implemented in the form we propose until now. We shall argue that this architecture has some of the advantages of decentralized supervision, where the IDS/IPS is located on a remote machine, connected to the monitored host through network links.

* This work was supported by grant DIGITEO N°2009-41D from Région Ile-de-France, and by project PFC ("plateforme de confiance"), pôle de compétitivité System@tic Paris-région Ile-de-France.

J. Garcia-Alfaro et al. (Eds.): DPM 2010 and SETOP 2010, LNCS 6514, pp. 244–258, 2011.

Outline. We survey related work in Section 2. Section 3 describes the rationale for our virtualized supervision architecture, and the architecture itself. We identify what we think is the weakest link in this architecture in Section 4, i.e., the need for domain zero supervision. There does not seem to be any silver bullet here. We attempt to explore a possible answer to the problem in Section 5, based on Sekar *et al.*'s *model-carrying code* paradigm. We conclude in Section 6.

2 Related Work

Much work has been done on enhancing the security of computer systems. Most implemented, host-based IDS run a program for security on the same operating system (OS) as protected programs and potential malware. This may be simply necessary, as with Janus [6], Systrace [13], Sekar *et al.*'s finite-state automaton learning system [19], or Piga-IDS [1], where the IDS must intercept and check each system call before execution. Call this an *interception* architecture: each system call is first checked for conformance against a security policy by the IDS; if the call is validated, then it is executed, otherwise the IDS forces the call to return immediately with an error code, without executing the call.

On the other hand, some other IDS are meant to work in a *decentralized* setting. In this case, the IDS does not run on the same host as the supervised host, S. While in an interception architecture, the IDS would run as a process on S itself, in a decentralized setting only a small so-called *sensor* running on S collects relevant events on S and sends them through some network link to the IDS, which runs on another, dedicated host M.

Both approaches have pros and cons. Interception architectures allow the IDS to counter any attack just before they are completed. This way, and assuming the security policy that the IDS enforces is sufficiently complete, no attack ever succeeds on S that would make reveal or alter sensitive data, make it unstable, or leave a backdoor open (by which we also include trojans and bots).

On the other hand, decentralized architectures (see Figure 1) make the IDS more resistant to attacks on S (which may be any of S_1, ..., S_4 in the figure): to kill the IDS, one would have to attack the supervision machine M, but M is meant to only execute the IDS and no other application, and has only limited network connectivity. In addition to the link from S to M used to report events to the IDS, we also usually have a (secure) link from M to S, allowing the IDS to issue commands to retaliate to attacks on S. While this may take time (e.g., some tens or hundreds of milliseconds on a LAN), this sometimes has the advantage to let the IDS *learn* about intruder behavior once they have completed an attack. This is important for forensics.

Decentralized architectures are also not limited to supervising just one host S. It is particularly interesting to let the supervision machine M collect events from several different hosts at once, from network equipment (routers, hubs, etc., typically through logs or MIB SNMP calls), and correlate between them, turning the IDS into a mix between host-based and network-based IDS.

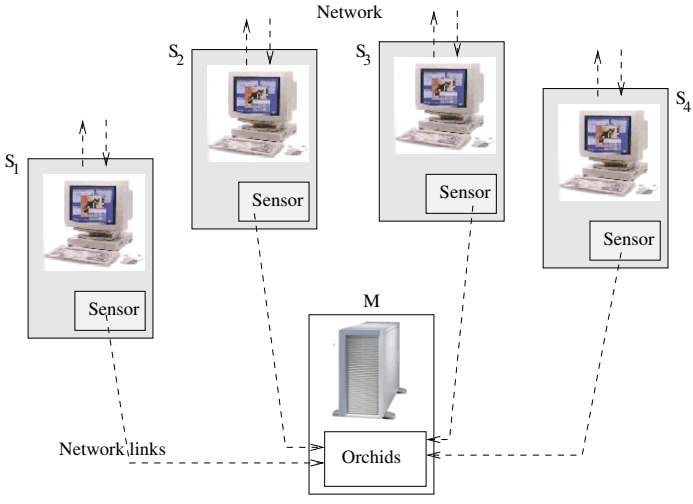


Fig. 1. Decentralized Supervision

We shall argue in Section 3 that one can simulate such a decentralized architecture, at minimal cost, on a single machine, using modern virtualization technology. We shall also see that this has some additional advantages.

A *virtualized* system such as Xen [28], VirtualBox [24], VMWare [25], or QEmu [15] allows one to emulate one or several so-called *guest* operating systems (OS) in one or several *virtual machines* (VM). The different VMs execute as though they were physically distinct machines, and can communicate through ordinary network connections (possibly emulated in software). The various VMs run under the control of a so-called *virtual machine monitor* (VMM) or *hypervisor*, which one can think of as being a thin, highly-privileged layer between the hardware and the VMs. See Figure 2, which is perhaps more typical of Xen than of the other cited hypervisors. The solid arrows are meant to represent the flow of control during system calls. When a guest OS makes a system call, its hardware layer is emulated through calls to the hypervisor. The hypervisor then calls the actual hardware drivers (or emulations thereof) implemented in a specific, high privilege VM called *domain zero*. Domain zero is the only VM to have access to the actual hardware, but is also responsible for administering the other VMs, in particular killing VMs, creating new VMs, or changing the emulated hardware interface presented to the VMs.

In recent years, virtualization has been seen by several as an opportunity for enforcing better security. The fact that two distinct VMs indeed run in separate sandboxes was indeed brought forward as an argument in this direction. However, one should realize that there is no conceptual distinction, from the point of view of protection, between having a high privilege VMM and lower-privileged VMs, and using a standard Unix operating system with a high privilege kernel and lower-privileged processes. Local-to-root exploits on Unix are bound to be imitated in the form of attacks that would allow one process running in one VM

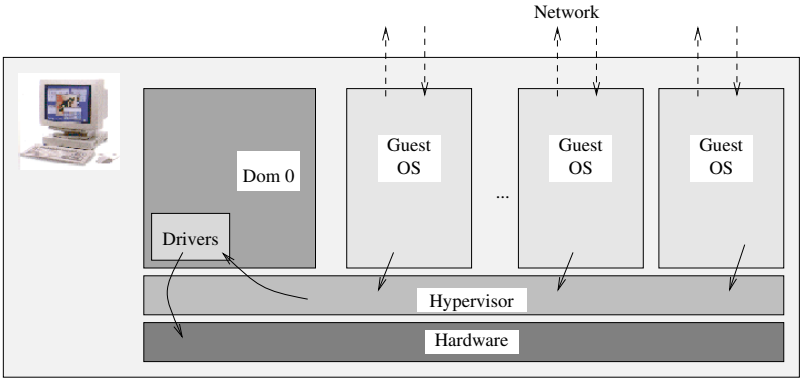


Fig. 2. A Virtualized System

to gain control of the full VMM, in particular of the full hardware abstraction layer presented to the VMs. Indeed, this is exactly what has started to appear, with Wojtczuk's attack notably [26].

Some of the recent security solutions using virtualization are sHype [18] and NetTop [10]. They provide infrastructure for controlling information flows and resource sharing between VMs. While the granularity level in these systems is a VM, our system controls execution at the granularity of a process.

Livewire [5] is an intrusion detection system that controls the behavior of a VM from the outside of the VM. Livewire uses knowledge of the guest OS to understand the behavior in a monitored VM. Livewire's VMM intercepts only write accesses to a non-writable memory area and accesses to network devices. On the other hand, our architecture can intercept and control all system calls invoked in target VMs.

In [12], Onoue *et al.* propose a security system that controls the execution of processes from the outside of VMs. It consists of a modified VMM and a program running in a trusted VM. The system intercepts system calls invoked in a monitored VM and controls the execution according to a security policy. Thus, this is a an interception system. To fill the semantic gap between low-level events and high-level behavior, the system uses knowledge of the structure of a given operating system kernel. The user creates this knowledge with a tool when recompiling the OS. In contrast, we do not need to rebuild the OS, and only need to rely on standard event-reporting daemons such as `auditd`, which comes with SELinux [22], but is an otherwise independent component.

We end this section by discussing run-time supervision and enforcement of security policies. Systems such as SELinux (op. cit.) are based on a security policy, but fail to recognize illegal *sequences* of legal actions. To give a simple example, it may be perfectly legal for user A to copy some private data D to some public directory such as `/tmp`, and for user B to read any data from `/tmp`, although our security policy forbids any (direct) flow of sensitive data

from A to B. Such sequences of actions are called *transitive flows* of data in the literature. Zimmerman *et al.* [29,30,31] propose an IDS that is able to check for illegal transitive flows. Briffaut [1] shows that even more general policies can be efficiently enforced, including non-reachability properties and Chinese Wall policies, among others; in general, Briffaut uses a simple and general policy language. We propose another, perhaps more principled, language in Section 5, based on linear temporal logic (LTL). Using the latter is naturally related to a more ancient proposal by Roger and the second author [17]. However, LTL as defined in (the first part of) the latter paper only uses future operators, and is arguably ill-suited to intrusion detection (as discussed in op. cit. already). Here, instead we use a fragment of LTL *with past*, which, although equivalent to ordinary LTL with only future operators as far as satisfiability is concerned (for some fixed initial state only, and up to an exponential-size blowup), will turn out to be much more convenient to specify policies, and easy to compile to rules that can be fed to the Orchids IPS [11,7].

3 Simulating Decentralized Supervision, Locally

We run a fast, modern IPS such as Orchids [11,7] in another VM to monitor, and react against, security breaches that may happen on the users' environment in each of the guest OSes present in a virtualized system: see Figure 3.

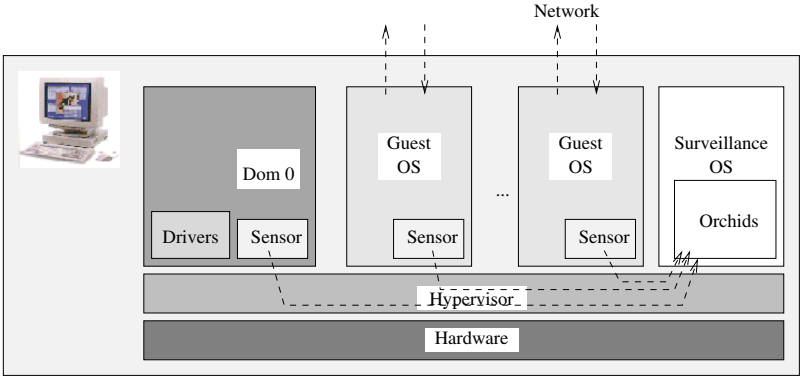


Fig. 3. Simulating Decentralized Supervision, Locally

One can see this architecture as an implementation of a decentralized supervision architecture on a single physical host.

We argue that this solution has several advantages. First, there is a clear advantage in terms of cost, compared to the decentralized architecture: we save the cost of the additional supervision host M.

Second, compared to a standard, unsupervised OS, the user does not need to change her usual environment, or to install any new security package. Only a small sensor has to run on her virtual machine to report events to Orchids.

Orchids accepts events from a variety of sensors. In our current implementation, each guest OS reports sequences of system calls through the standard `auditd` daemon, a component of SELinux [22], which one can even run without the need for installing or running SELinux itself. (Earlier, we used Snare, however this now seems obsolescent.) The bulk of the supervision effort is effected in a different VM, thus reducing the installation effort to editing a few configuration files, to describe the connections between the guest OSes and the supervision OS mainly. In particular, we do not need to recompile any OS kernel with our architecture, except possibly to make sure that `auditd` is installed and activated.

A third advantage, compared with interception architectures, and which we naturally share with decentralized architectures, is that isolating the IPS in its own VM makes it resistant to attacks from the outside. Indeed, Orchids runs in a VM that has no other network connection to the outside world than those it requires to monitor the guest OSes, and which runs no other application that could possibly introduce local vulnerabilities.

Orchids should have high privileges to be able to retaliate to attacks on each guest OS. For example, we use `ssh` connections between Orchids and each VM kernel to be able to kill offending processes or disable offending user accounts. (The necessary local network links, running in the opposite direction as the sensor-to-Orchids event reporting links shown in Figure 3, are not drawn.)

However, running on a virtualized architecture offers additional benefits. One of them is that Orchids can now ask domain zero to *kill* an entire VM. This is necessary when a guest OS has been subject to an attack with consequences that we cannot assess precisely. For example, the `do_brk()` attack [23] and its siblings, or the `vmsplice()` attack [14] allow the attacker not just to gain root access, but direct access to the *kernel*. Note that this means, for example, that the attacker has immediate access to the whole process table, as well as to the memory zones of all the processes. While current exploits seem not to have used this opportunity, such attack vectors in principle allow an attacker to become completely stealthy, e.g., by making its own processes invisible to the OS. In this case, the OS is essentially in an unpredictable state.

The important point is that we can always revert any guest OS to a previous, safe state, using virtualization. Indeed, each VM can be *checkpointed*, i.e., one can save the complete instantaneous state of a given VM on disk, including processes, network connections, signals. Assuming that we checkpoint each VM at regular intervals, it is then feasible to have Orchids retaliate by killing a VM in extreme cases and replacing it by an earlier, safe checkpoint.

Orchids can also detect VMs that have died because of fast denial-of-service attacks (e.g., the double `listen()` attack [3], which causes instant kernel lock-up), by pinging each VM at regular intervals: in this case, too, Orchids can kill the VM and reinstall a previous checkpoint. We react similarly to attacks on guest OSes that are suspected of having succeeded in getting kernel privileges and of, say, disabling the local `auditd` daemon.

Killing VMs and restoring checkpoints is clearly something that we cannot afford with physical hosts instead of VMs.

It would be tempting to allow Orchids to run *inside* domain zero to do so. Instead, we run Orchids in a separate guest OS, with another `ssh` connection to issue VM administration commands to be executed by a shell in domain zero. I.e., we make domain zero *delegate* surveillance to a separate VM running Orchids, while the latter trusts domain zero to administer the other guest VMs. We do so in order to sandbox each from the other one. Although we have taken precautions against this prospect, there is still a possibility that a wily attacker would manage to cause denial-of-service attacks on Orchids by crafting events causing blow-up in the internal Orchids surveillance algorithm (see [7]), and we don't want this to cause the collapse of the whole host. Conversely, if domain zero itself is under attack, we would like Orchids to be able to detect this and react against it. We discuss this in Section 4.

To our knowledge, this simple architecture has not been put forward in previous publications, although some proposals already consider managing the security of virtualized architectures, as we have already discussed in Section 2.

4 The Weak Link: Domain Zero Administration

The critical spots in our architecture are the VMM (hypervisor) itself, domain zero, and the surveillance VM running Orchids.

Attacking the latter is a nuisance, but is not so much of a problem as attacking the VMM or domain zero, which would lead to complete subversion of the system. Moreover, the fact that Orchids runs in an isolated VM averts most of the effects of any vulnerability that Orchids may have.

Attacks against the VMM are much more devastating. Wojtczuk's 2008 attacks on Xen 2 [26] allow one to take control of the VMM, hence of the whole machine, by rewriting arbitrary code and data using DMA channels, and almost without the processor's intervention... quite a fantastic technique, and certainly one that breaks the common idea that every change in stored code or data must be effected by some program running on one of the processors. Indeed, here a separate, standard chip is actually used to rewrite the code and data.

Once an attacker has taken control over the VMM, one cannot hope to react in any effective way. In particular, the VMM controls entirely the hardware abstraction layer that is presented to each of the guest OSes: no network link, no disk storage facility, no keyboard input can be trusted by any guest OS any longer. Worse, the VMM also controls some of the features of the processor itself, or of the MMU, making memory or register contents themselves unreliable.

We currently have no idea how to prevent attacks such as Wojtczuk's, apart from unsatisfactory, temporary remedies such as checkpointing some selected memory areas in the VMM code and data areas.

However, we claim that averting such attacks is best done by making sure that they cannot be run at all. Indeed, Wojtczuk's attacks only work provided the attacker already has *root access* to domain zero, and this is already quite a predicament. We therefore concentrate on ensuring that no unauthorized user can gain root access to domain zero.

Normally, only the system administrator should have access to domain zero. (In enterprise circles, the administrator may be a person with the specific role of an administrator. In family circles, we may either decide that there should be no administrator, and that the system should self-administer; or that one particular user has administrator responsibilities.) We shall assume that this administrator is *benevolent*, i.e., will not consciously run exploits against the system. However, he may do so without knowing it.

One possible scenario is this: the administrator needs to upgrade some library or driver, and downloads a new version; this version contains a trojan horse, which runs Wojtczuk's attack. Modern automatic update mechanisms use cryptographic mechanisms, including certificates and cryptographic hashing mechanisms [27], to prevent attackers from running man-in-the-middle attacks, say, and substitute a driver with a trojan horse for a valid driver. However, there is no guarantee that the authentic driver served by the automatic update server is itself free of trojans. There is at least one actual known case of a manufacturer shipping a trojan (hopefully by mistake): some Video iPods were shipped with the Windows `RavMonE.exe` virus [16], causing immediate infection of any Windows host to which the iPods were connected.

5 A Line of Defense: MCC, LTL, and Orchids

Consider the case whereby we have just downloaded a device driver, and we would like to check whether it is free of a trojan. Necula and Lee pioneered the idea of shipping a device driver with a small, formal proof of its properties, and checking whether this proof is correct before installing and running the driver. This is *proof-carrying code* [9]. More suited to security, and somehow more practical is Sekar *et al.*'s idea of *model-carrying code* (MCC) [20]. Both techniques allow one to accept and execute code even from untrusted producers.

Let us review MCC quickly. A *producer* generates both the program D to be downloaded (e.g., the device driver), and a *model* of it, M. The *consumer*, which in our case is software running in domain zero, checks the model against a local *policy* P. Instead of merely rejecting the program D if its model M does not

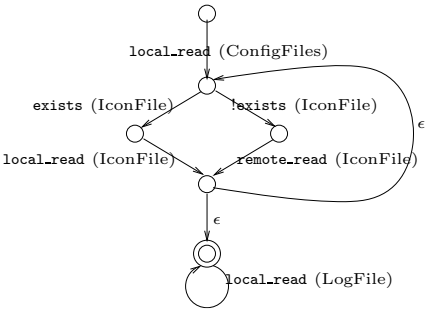


Fig. 4. An EFSA Model, after Sekar *et al.* [20]

satisfy, the consumer computes an *enforcement model* M' that satisfies both M and P, and generates a monitor that checks whether M' satisfies P at run-time. Any violation is flagged and reported.

In [20], models, as well as policies and enforcement models, are taken to be extended finite-state automata (EFSA), i.e., finite state automata augmented with finitely many state variables meant to hold values over some fixed domain. A typical example, taken from op. cit., is the EFSA of Figure 4. This is meant to describe the normal executions of D as doing a series of system calls as follows. Each system call is abstracted, e.g., the first expected system call from D is a call to `local_read` with argument bound to the ConfigFiles state variable. Then D is expected to either take the left or the right transition going down, depending on whether some tested file (in variable IconFile) exists or not. In the first case, D should call `local_read`, in the second case, D should call `remote_read`. The transitions labeled ϵ are meant to be firable spontaneously.

Typical policies considered by Sekar *et al.* are invariants of the form "any program should either access local files or access the network but not both" (this is violated above, assuming obvious meanings for system calls), or that only files from the `/var/log/httpd/` directory should be read by D. Policies are again expressed as EFSA, and enforcement models can be computed as a form of synchronized product denoting the intersection of the languages of M and P.

The EFSA model is sufficiently close to the automaton-based model used in Orchids (which appeared about at the same time, see the second part of [17]; or see [7] for a more in-depth treatment) that the EFSA built in the MCC approach can be fed almost directly to Orchids. Accordingly, we equip domain zero with a sensor as well (Figure 3, left box), which reports to Orchids for EFSA checking.

However, we feel that a higher-level language, allowing one to write acceptable policies for automatic updates in a concise and readable manner, would be a plus. There have been many proposals of higher-level languages already, including linear temporal logic (LTL) [17], chronicles [8], or the BMSL language by Sekar and Uppuluri [21], later improved upon by Brown and Ryan [2]. It is not our purpose to introduce yet another language here, but to notice that a simple variant of LTL *with past* will serve our purpose well and is efficiently and straightforwardly translated to Orchids rules—which we equate with EFSA here, for readability, glossing over inessential details.

Consider the following fragment of LTL with past. We split the formulae in several sorts. $F^\bullet$ will always denote *present tense* formulae, which one can evaluate by just looking at the current event:

$$\begin{array}{rcll} F^\bullet & ::= & P(\boldsymbol{x}) \mid \mathit{cond}(\boldsymbol{x}) & \text{atomic formula} \\ & \mid & \bot & \text{false} \\ & \mid & F^\bullet \wedge F^\bullet & \text{conjunction} \\ & \mid & F^\bullet \vee F^\bullet & \text{disjunction} \end{array}$$

Atomic formulae check for specific occurrences of events, e.g., `local_read` (IconFile) will typically match the current event provided it is of the form `local_read` applied to some argument, which is bound to the state variable IconFile. In the

above syntax, $\boldsymbol{x}$ denotes a list of state variables, while $cond(\boldsymbol{x})$ denotes any computable formula of $\boldsymbol{x}$, e.g., to check that IconFile is a file in some specific set of allowed directories. This is as in [21,2]. We abbreviate $P(\boldsymbol{x}) \mid \top$, where $\top$ is some formula denoting true, as $P(\boldsymbol{x})$.

Note that we do not allow for negations in present tense formulae. If needed, we allow certain negations of atomic formulae as atomic formulae themselves, e.g., `!exists` (IconFile). However, we believe that even this should not be necessary. Disjunctions were missing in [21], and were added in [2].

Next, we define *past tense* formulae, which can be evaluated by looking at the current event and all past events, but none of the events to come. Denote past tense formulae by $F^{\leftarrow}$:

$$\begin{array}{rcll} F^{\leftarrow} & ::= & F^{\bullet} & \text{present tense formulae} \\ & \mid & F^{\leftarrow} \wedge F^{\leftarrow} & \text{conjunction} \\ & \mid & F^{\leftarrow} \vee F^{\leftarrow} & \text{disjunction} \\ & \mid & F^{\leftarrow} \setminus F^{\bullet} & \text{without} \\ & \mid & \text{Start} & \text{initial state} \end{array}$$

All present formulae are (trivial) past formulae, and past formulae can also be combined using conjunction and disjunction. The novelty is the "without" constructor: $F^{\leftarrow} \setminus F^{\bullet}$ holds iff $F^{\leftarrow}$ held at some point in the past, and since then, $F^{\bullet}$ never happened. Apart from the without operator, the semantics of our logic is standard. We shall see below that it allows us to encode a number of useful idioms. The past tense formula Start will also be explained below.

Formally, present tense formulae $F^{\bullet}$ are evaluated on a current event e, while past tense formulae $F^{\leftarrow}$ are evaluated on a stream of events $\boldsymbol{e} = e_1, e_2, \ldots, e_n$, where the current event is e_n, and all others are the past events. (We warn the reader that the semantics is meant to reason logically on the formulae, but is not indicative of the way they are evaluated in practice. In particular, although we are considering past tense formulae, and their semantics refer to past events, our algorithm will never need to read back past events.) The semantics of the without operator is that $\boldsymbol{e} = e_1, e_2, \ldots, e_n$ satisfies $F^{\leftarrow} \setminus F^{\bullet}$ if and only if there is an integer m, with $0 \leq m < n$, such that the proper prefix of events $e_1, e_2, \ldots, e_m$ satisfies $F^{\leftarrow}$ for some values of the variables that occur in $F^{\leftarrow}$ ("$F^{\leftarrow}$ held at some point in the past"), and none of $e_{m+1}, \ldots, e_n$ satisfies $F^{\bullet}$ ("since then, $F^{\bullet}$ never happened")—precisely, none of $e_{m+1}, \ldots, e_n$ satisfies $F^{\bullet}$ *with the values of the variables obtained* so as to satisfy $F^{\leftarrow}$; this makes perfect sense if all the variables that occur in $F^{\leftarrow}$ already occur in $F^{\bullet}$, something we shall now assume.

The past tense formula Start has trivial semantics: it only holds on the empty sequence of events (i.e., when $n = 0$), i.e., it only holds when we have not received any event yet. This is not meant to have any practical use, except to be able to encode useful idioms with only a limited supply of temporal operators. For example, one can define the formula $\blacksquare\neg F^{\bullet}$ ("$F^{\bullet}$ never happened in the past") as Start $\setminus F^{\bullet}$.

The without operator allows one to encode other past temporal modalities, see Figure 5. In particular, we retrieve the *chronicle* $F_1^{\bullet}; F_2^{\bullet}; \ldots; F_n^{\bullet}$ [8], meaning that events matching $F_1^{\bullet}$, then $F_2^{\bullet}$, ..., then $F_n^{\bullet}$ have occurred in this

$$\blacksquare\neg F^\bullet \stackrel{\text{def}}{=} \text{Start} \diagdown F^\bullet \qquad \text{“}F^\bullet \text{ never happened in the past”}$$
$$\blacklozenge F^{\leftarrow} \stackrel{\text{def}}{=} F^{\leftarrow} \diagdown \bot \qquad \text{“}F^{\leftarrow} \text{ was once true in the past”}$$
$$F^{\leftarrow} \to F^\bullet \stackrel{\text{def}}{=} \blacklozenge F^{\leftarrow} \wedge F^\bullet \qquad \text{“}F^{\leftarrow} \text{ was once true, and now } F^\bullet \text{ is”}$$
$$F^{\leftarrow} \to F_1^\bullet \to F_2^\bullet \to \ldots \to F_n^\bullet \stackrel{\text{def}}{=} (\ldots((F^{\leftarrow} \to F_1^\bullet) \to F_2^\bullet) \to \ldots) \to F_n^\bullet$$
$$F_1^\bullet; F_2^\bullet; \ldots; F_n^\bullet \stackrel{\text{def}}{=} \text{Start} \to F_1^\bullet \to \ldots \to F_n^\bullet \text{ Chronicle}$$

Fig. 5. Some Useful Idioms

order before, not necessarily in a consecutive fashion. More complex sequences can be expressed. Notably, one can also express disjunctions as in [2], e.g., disjunctions of chronicles, or formulae such as $(\texttt{login}(Uid) \diagdown \texttt{logout}(Uid)) \wedge \texttt{local\_read}(Uid, ConfigFile)$ to state that user Uid logged in, then read some $ConfigFile$ locally, without logging out inbetween.

Let us turn to more practical details. First, we do not claim that only Start and the without ($\diagdown$) operator should be used. The actual language will include syntactic sugar for chronicles, box ($\blacksquare$) and diamond ($\blacklozenge$) modalities, and possibly others, representing common patterns. The classical past tense LTL modality $\mathcal{S}$ (“since”) is also definable, assuming negation, by $F \mathcal{S} G = G \diagdown \neg F$, but seems less interesting in a security context.

Second, as already explained in [17,21,2], we see each event e as a formula $P(fld_1, fld_2, \ldots, fld_m)$, where $fld_1, fld_2, \ldots, fld_m$ are taken from some domain of values—typically strings, or integers, or time values. This is an abstraction meant to simplify mathematical description. For example, using `auditd` as event collection mechanism, we get events in the form of strings such as:

```
1276848926.326:1234 syscall=102 success=yes a0=2 a1=1 a2=6 pid=7651
```

which read as follows: the event was collected at date 1276848926.326, written as the number of seconds since the epoch (January 01, 1970, 0h00 UTC), and is event number 1234 (i.e., we are looking at event e_{1234} is our notation); this was a call to the `socket()` function (code 102), with parameters `PF_INET` (Internet domain, where `PF_INET` is defined as 2 in `/usr/include/socket.h`—`a0` is the first parameter to the system call), `SOCK_STREAM` (= 1; `a1` is connection type here), and with the TCP protocol (number 6, passed as third argument `a2`); this was issued by process number 7651 and returned with success. Additional fieldss that are not relevant to the example are not shown. This event will be understood in our formalization as event e_{1234}, denoting `syscall` $(1276848926.326, 102, \texttt{"yes"}, 2, 1, 6, 7651)$. The event e_{1234} satisfies the atomic formula `syscall` $(Time, Call, Res, Dom, Conn, Prot, Pid) \mid Res = \texttt{"yes"}$ but neither `audit` (X) nor `syscall` $(Time, Call, Res, Dom, Conn, Prot, Pid) \mid Time \leq 1276848925$.

Third, we need to explain how we can detect when a given sequence of events $\boldsymbol{e}$ satisfies a given formula in our logic, algorithmically. To this end, we define a translation to the Orchids language, or to EFSA, and rely on Orchids' extremely efficient model-checking engine [7]. The translation is based on the idea of *history variables*, an old idea in model-checking safety properties in propositional

LTL. Our LTL is *not* propositional, as atomic formulae contain free variables—one may think of our LTL as being first-order, with an implicit outer layer of existential quantifiers on all variables that occur—but a similar technique works.

It is easier to define the translation for an extended language, where the construction $F^{\leftarrow} \setminus F^{\bullet}$ is supplemented with a new construction $F^{\leftarrow} \setminus^* F^{\bullet}$ (*weak without*), which is meant to hold iff $F^{\leftarrow}$ once held in the past, *or holds now*, and $F^{\bullet}$ did not become true afterwards.

The *subformulae* of a formula F are defined as usual, as consisting of F plus all subformulae of its immediate subformulae. To avoid some technical subtleties, we shall assume that Start is also considered a subformula of any past tense formula. The *immediate subformulae* of $F \wedge G$, $F \vee G$, $F \setminus^* G$ are F and G, while atomic formulae, $\bot$ and Start don't have any immediate subformula. To make the description of the algorithm smoother, we shall assume that the immediate subformulae of $F \setminus G$ are not F and G, but rather $F \setminus^* G$ and G. Indeed, we are reproducing a form of Fischer-Ladner closure here [4].

Given a fixed past-tense formula $F^{\leftarrow}$, we build an EFSA that monitors exactly when a sequence of events will satisfy $F^{\leftarrow}$. To make the description of the algorithm simpler, we shall assume a slight extension of Sekar *et al.*'s EFSA where state variables can be assigned values on traversing a transition. Accordingly, we label the EFSA transitions with a sequence of *actions* $\$x_1 := e_1; \$x_2 := e_2; \ldots; \$x_k := e_k$, where $\$x_1$, $\$x_2$, …, $\$x_k$ are state variables, and e_1, e_2, …, e_k are expressions, which may depend on the state variables. This is actually possible in the Orchids rule language, although the view that is given of it in [7] does not mention it. Also, we will only need these state variables to have two values, 0 (false) or 1 (true), so it is in principle possible to dispense with all of them, encoding their values in the EFSA's finite control. (Instead of having three states, the resulting EFSA would then have $3\ 2^k$ states.)

Given a fixed $F^{\leftarrow}$, our EFSA has only three states q_{init} (the initial state), q, and q_{alert} (the final, acceptance state). We create state variables $\$x_i$, $1 \leq i \leq k$, one per subformula of $F^{\leftarrow}$. Let F_1, F_2, …, F_k be these subformulae (present or past tense), and sort them so that any subformula of F_i occurs before F_i, i.e., as F_j for some $j < i$. (This is a well-known *topological sort*.) In particular, F_k is just $F^{\leftarrow}$ itself. Without loss of generality, let Start occur as F_1. The idea is that the EFSA will run along, monitoring incoming events, and updating $\$x_i$ for each i, in such a way that, at all times, $\$x_i$ equals 1 if the corresponding subformula F_i holds on the sequence $\boldsymbol{e}$ of events already seen, and equals 0 otherwise.

There is a single transition from q_{init} to q, which is triggered without having to read any event at all. This is an ϵ-*transition* in the sense of [7], and behaves similarly to the transitions `exists` (IconFile) and `!exists` (IconFile) of Figure 4. It is labeled with the actions $\$x_1 := 1; \$x_2 := 0; \ldots; \$x_k := 0$ (Start holds, but no other subformula is currently true).

There is also a single ϵ-transition from q to q_{alert}. This is labeled by no action at all, but is guarded by the condition $\$x_k == 1$. I.e., this transition can only be triggered if $\$x_k$ equals 1. By the discussion above, this will only ever happen when F_k, i.e., $F^{\leftarrow}$ becomes true.

Finally, there is a single (non-ϵ) transition from q to itself. Since it is not an ϵ-transition, it will only fire on reading a new event e [7]. It is labeled with the following actions, written in order of increasing values of i, $1 \leq i \leq k$:

$\$x_1 := 0$	(Start is no longer true)
$\$x_i := P(\boldsymbol{x}) \wedge cond(\boldsymbol{x})$	(for each i such that F_i is atomic, i.e., F_i is $P(\boldsymbol{x}) \mid cond(\boldsymbol{x})$)
$\$x_i := 0$	(if F_i is $\bot$)
$\$x_i := and(\$x_j, \$x_k)$	(if $F_i = F_j \wedge F_k$)
$\$x_i := or(\$x_j, \$x_k)$	(if $F_i = F_j \vee F_k$)
$\$x_i := or(\$x_j, and(not(\$x_k), \$x_i))$	(if $F_i = F_j \nwarrow^* F_k$)
$\$x_i := and(not(\$x_k), \$x_\ell)$	(if $F_i = F_j \nwarrow F_k$, and $F_j \nwarrow^* F_k$ is F_ℓ, $\ell < i$)

Here, *and*, *or* and *not* are truth-table implementations of the familiar Boolean connectives, e.g., $and(0,1)$ equals 0, while $and(1,1)$ equals 1. We assume that $P(\boldsymbol{x})$, i.e., $P(x_1, \ldots, x_n)$ will equal 1 if the current event is of the form $P(s_1, \ldots, s_n)$, and provided each x_j that was already bound was bound to s_j exactly, in which case those variable x_j that were still unbound will be bound to the corresponding s_j. E.g., if x_1 is bound to 102 but x_2 is unbound, then $P(x_1, x_2)$ will equal 1 if the current event is $P(102, 6)$ (binding x_2 to 6), or $P(102, 7)$ (binding x_2 to 7), but will equal 0 if the current event is $Q(102, 6)$ for some $Q \neq P$, or $P(101, 6)$. We hope that this operational view of matching predicates is clearer than the formal view (which simply treats $x_1, \ldots, x_n$ as existentially quantified variables, whose values will be found as just described).

The interesting case is when F_i is a without formula $F_j \nwarrow F_k$, or $F_j \nwarrow^* F_k$. $F_j \nwarrow F_k$ will become true after reading event e whenever $F_j \nwarrow^* F_k$ was already true before reading it, and F_k is still false, i.e., when $\$x_\ell = 1$ and $\$x_k = 0$, where ℓ is the index such that $F_j \nwarrow^* F_k$ occurs in the list of subformulae of $F^{\leftarrow}$ as F_ℓ. So in this case we should update $\$x_i$ to $and(not(\$x_k), \$x_\ell)$, as shown above. This relies on updating variables corresponding to weak without formulae $F_j \nwarrow^* F_k$: $F_j \nwarrow^* F_k$ becomes true after reading event e iff either F_j becomes true ($\$x_j = 1$), or $F_j \nwarrow^* F_k$ was already true before ($\$x_i$ was already equal to 1) and F_k is false on event e ($\$x_k$ equals 0), whence the formula $\$x_i := or(\$x_j, and(not(\$x_k), \$x_i))$ in this case.

This completes the description of the translation. We now rely on Orchids' fast, real-time monitoring engine to alert us in case any policy violation, expressed in our fragment of LTL, is detected.

6 Conclusion

We have proposed a cost-effective implementation of a decentralized supervision architecture on a single host, using virtualization techniques. This offers at least the same security level as a classical decentralized intrusion detection architecture based on a modern IPS such as Orchids, and additionally allows for killing and restoring whole VMs to previous safe states.

In any security architecture, there is a weak link. We identify this weak link as domain zero administration, where even a benevolent administrator may introduce trojans, possibly leading to complete corruption, not just of the VMs, but of the VMM itself. There is no complete defense against this yet. As a first step, we have shown that Orchids can again be used, this time to implement an elegant instance of Sekar *et al.*'s model-carrying code paradigm (MCC).

References

1. Briffaut, J.: Formalisation et garantie de propriétés de sécurité système: Application à la détection dintrusions. PhD thesis, LIFO Université d'Orléans, ENSI Bourges (December 2007)
2. Brown, A., Ryan, M.: Synthesising monitors from high-level policies for the safe execution of untrusted software. In: Chen, L., Mu, Y., Susilo, W. (eds.) ISPEC 2008. LNCS, vol. 4991, pp. 233–247. Springer, Heidelberg (2008)
3. Dias, H.: Linux kernel 'net/atm/proc.c' local denial of service vulnerability. BugTraq Id 32676, CVE-2008-5079 (December 2008)
4. Fischer, M.J., Ladner, R.E.: Propositional dynamic logic of regular programs. Journal of Computer and System Sciences 18, 194–211 (1979)
5. Garfinkel, T., Rosenblum, M.: A virtual machine introspection based architecture for intrusion detection. In: Proceedings of the 10th Annual Network and Distributed Systems Security Symposium, San Diego, CA (February 2003)
6. Goldberg, I., Wagner, D., Thomas, R., Brewer, E.A.: A secure environment for untrusted helper applications (confining the wily hacker). In: Proceedings of the 6th USENIX Security Symposium, San Jose, CA (July 1996)
7. Goubault-Larrecq, J., Olivain, J.: A smell of ORCHIDS. In: Leucker, M. (ed.) RV 2008. LNCS, vol. 5289, pp. 1–20. Springer, Heidelberg (2008)
8. Morin, B., Debar, H.: Correlation of intrusion symptoms: an application of chronicles. In: Vigna, G., Krügel, C., Jonsson, E. (eds.) RAID 2003. LNCS, vol. 2820, pp. 94–112. Springer, Heidelberg (2003)
9. Necula, G.C., Lee, P.: Safe kernel extensions without run-time checking. SIGOPS Operating Systems Review 30, 229–243 (1996)
10. NetTop (2004), `http://www.nsa.gov/research/tech_transfer/fact_sheets/nettop.shtml`
11. Olivain, J., Goubault-Larrecq, J.: The ORCHIDS intrusion detection tool. In: Etessami, K., Rajamani, S.K. (eds.) CAV 2005. LNCS, vol. 3576, pp. 286–290. Springer, Heidelberg (2005)
12. Onoue, K., Oyama, Y., Yonezawa, A.: Control of system calls from outside of virtual machines. In: Wainwright, R.L., Haddad, H. (eds.) SAC, pp. 2116–1221. ACM, New York (2008)
13. Provos, N.: Improving host security with system call policies. In: Proceedings of the 12th USENIX Security Symposium, Washington, DC (August 2003)
14. Purczyński, W., qaaz: Linux kernel prior to 2.6.24.2 'vmsplice_to_pipe()' local privilege escalation vulnerability (February 2008), `http://www.securityfocus.com/bid/27801`
15. Qemu (2010), `http://www.qemu.org/`
16. Small number of video iPods shipped with Windows virus (2010), `http://www.apple.com/support/windowsvirus/`

17. Roger, M., Goubault-Larrecq, J.: Log auditing through model checking. In: 14th IEEE Computer Security Foundations Workshop (CSFW 2001), pp. 220–236. IEEE Comp. Soc. Press, Los Alamitos (2001)
18. Sailer, R., Jaeger, T., Valdez, E., Caceres, R., Perez, R., Berger, S., Griffin, J., Doorn, L.: Building a MAC-based security architecture for the Xen opensource hypervisor. In: Proceedings of the 21st Annual Computer Security Applications Conference, Tucson, AZ (December 2005)
19. Sekar, R., Bendre, M., Bollineni, P., Dhurjati, D.: A fast automaton-based method for detecting anomalous program behaviors. In: IEEE Symposium on Security and Privacy, Oakland, CA (May 2001)
20. Sekar, R., Ramakrishnan, C., Ramakrishnan, I., Smolka, S.: Model-carrying code (MCC): A new paradigm for mobile-code security. In: Proceedings of the New Security Paradigms Workshop (NSPW 2001). ACM Press, Cloudcroft (September 2001)
21. Sekar, R., Uppuluri, P.: Synthesizing fast intrusion prevention/detection systems from high-level specifications. In: Proceedings of the 8th Conference on USENIX Security Symposium, SSYM 1999, Berkeley, CA (1999)
22. Smalley, S., Vance, C., Salamon, W.: Implementing SELinux as a Linux security module. Technical report, NSA (2001)
23. Starzetz, P.: Linux kernel 2.4.22 do_brk() privilege escalation vulnerability. K-Otik ID 0446, CVE CAN-2003-0961 (December 2003), `http://www.k-otik.net/bugtraq/12.02.kernel.2422.php`
24. Virtualbox (2010), `http://www.virtualbox.org/`
25. Vmware (2010), `http://www.vmware.com/`
26. Wojtczuk, R.: Subverting the Xen hypervisor. In: Black Hat 2008, Las Vegas, NV (2008)
27. [ms-wusp]: Windows update services: Client-server protocol specification (2007-2010), `http://msdn.microsoft.com/en-us/library/cc251937PROT.13.aspx`
28. Xen (2005-2010), `http://www.xen.org/`
29. Zimmermann, J., Mé, L., Bidan, C.: Introducing reference flow control for detecting intrusion symptoms at the OS level. In: Wespi, A., Vigna, G., Deri, L. (eds.) RAID 2002. LNCS, vol. 2516, pp. 292–306. Springer, Heidelberg (2002)
30. Zimmerman, J., Mé, L., Bidan, C.: Experimenting with a policy-based hids based on an information flow control model. In: ACSAC 2003: Proceedings of the 19th Annual Computer Security Applications Conference, Washington, DC, USA, p. 364. IEEE Computer Society, Los Alamitos (2003)
31. Zimmermann, J., Mé, L., Bidan, C.: An improved reference flow control model for policy-based intrusion detection. In: Snekkenes, E., Gollmann, D. (eds.) ESORICS 2003. LNCS, vol. 2808, pp. 291–308. Springer, Heidelberg (2003)

Author Index