A Model-Checking Approach for Enforcing Purpose-based Privacy Policies

Rezvan Joshaghani
Department of Computer Science
College of Engineering
Boise State University
Boise, Idaho 83725
Email: RezvanJoshaghani@u.boisestate.edu

Hoda Mehrpouyan
Department of Computer Science
College of Engineering
Boise State University
Boise, Idaho 83725
Email: HodaMehrpouyan@boisestate.edu

***Abstract*—With the growth of Internet in many different aspects of life, users are required to share private information more than ever. Hence, users need a privacy management tool that can enforce complex and customized privacy policies. In this paper, we propose a privacy management system that not only allows users to define complex privacy policies for data sharing actions, but also monitors users' behavior and relationships to generate realistic policies. In addition, the proposed system utilizes formal modeling and model-checking approach to prove that information disclosures are valid and privacy policies are consistent with one another.**

1. Introduction

With the growth of Internet businesses, IoT and smart devices the amount of data that we are sharing with others everyday is considerable and raises many privacy issues. In the information system and software engineering domain, privacy protection represents the capability that the individual controls the collection, exposition and maintenance of information about themselves [1]. However, different devices require different privacy configurations, usually these configurations are not customized according to the user's preferences. To address this issue, a privacy management system is proposed that not only enables the user to have customized privacy policies, but also takes into the consideration the continuous changing of privacy requirements and dynamic nature of the relationships and circumstances.

In order to verify that the discloser of information is in compliance with the policies, model checking is used. Model checking is an automatic technique for verifying finite state concurrent systems [2]. In this research, model checking is prefered over other methods such as rule-based systems because it has the ability to exhaustively check all possible paths in which information disclouser might accure . In this case, privacy policy model is created from user's privacy requirements which is then verified against the data sharing action to make sure that no policy is violated or against a newly generated policy to make sure that all policies are consistan. If a violation happens, our system gives feedback to the user and if there was no violation of the policies, the system will let the action happen or adds a new policy to the policy model. The fact that the privacy policy management field is rather complex and an open research area, studying related works is highly advantageous and of importance. As a result the following section will take a brief look on two of the most related and used works.

2. Related Works

Given the growing awareness of identity theft by the society and other misuses of personal information, it is not surprising that privacy remains a very active area of research for current and emerging technology design [3].In [1] the authors provided a method for checking the privacy policies within each web service using Spin model checker to see whether the privacy policies have been violated or not. In comparison, our approach is toward each and every user so that there is a unique policy for each individual. Moreover, our user's privacy policies are dynamic as opposed to the static privacy policies used by web services.

In order to utilize model checking on privacy policies machine readable policies are of necessity where in [3] the authors created a policy workbench called SPARCLE to transform natural language privacy policies to machine readable ones. SPARCLE gets the policies in natural language and generates a XML version of the policies. They have created a set of grammars which execute on a shallow parser that are designed to identify the rule elements in privacy policy rules. Inspired by their work and aiming for an even more complex policies, new categories such as time and gateway were added.

3. Methodology

This section is about the structure of the privacy management system, first there is an overview on how the privacy management system works in general, then there is an explanation of each part in particular.
As depicted in Figure1, an initial set of privacy policies are defined by the user and given to the privacy management system to create the privacy policy models. Further, the user's behavior for any data sharing action are monitored, i.e. the number of emails that are sent to a particular friend, either represent a new friendship or a dissolved friendship. If a data sharing action or policy change is detected, the system checks the validation of the action or the new policy, against the privacy model. If the action or the new policy violates the privacy models of the system, the user is informed of this violation and the user will decide on whether or not

978-1-5386-1027-5/17 $31.00 © 2017 IEEE
DOI 10.1109/PAC.2017.31

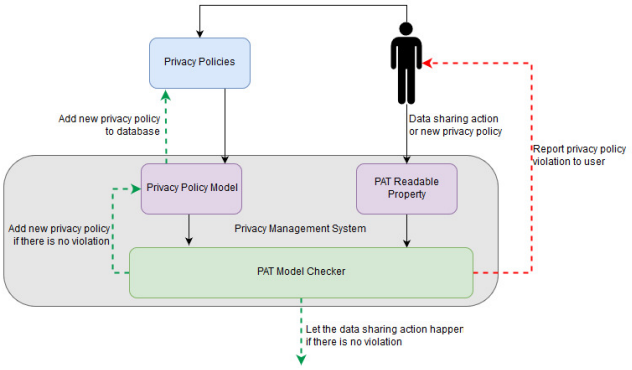

Figure 1. System overview.

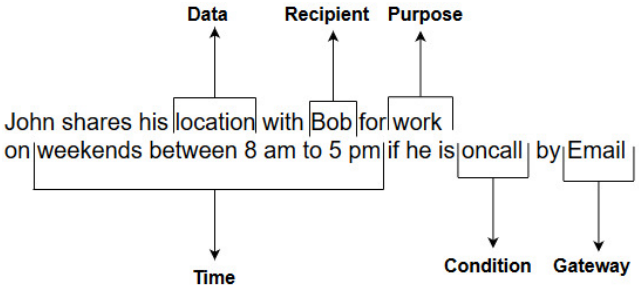

Figure 2. Example of privacy policy structure.

to perform the action or choose between the new and old policies. In cases that there are no violations or the user decides to ignore them, the system lets the data sharing action happen.

3.1. Privacy Policy Structure

In this research, a machine readable structure is designed, which breaks down each policy to categories such as Recipient: people, organizations, applications, web sites, etc. that a user sends information to. Data: The pieces of information that a user can share with different recipients. Purpose: The allowed reasons a recipient can receive a particular kind of data. Conditions: The conditions that should be met before sharing information. Time: For the policies that should be applied in specific date or time. Gateway: For the information that should be applied through a specific application,sensor,etc. Figure 2 is an example of above structure.

3.2. Model Checking

By model checking the information flow of the system against the policy model, it is possible to make sure that the data sharing action is not violating any policies and the policies always stay consistent.

3.2.1. PAT Model Checker. In this research, in order to verify that each action or new policy complies with the privacy policy model, PAT model checker is used, because of it support of other model checking features such as probabilistic and real time model checking [4].

In PAT, a process is defined as an equation in the following syntax,$P(x1, x2, ..., xn) = Exp$ where P is the process name, and $x1, ..., xn$ is an optional list of process parameters. Further, Exp is defined as a process expression, which determines the computational logic of the process. PAT offers different operators on processes and one of them is general choice, that is defined as $P[]Q$. The choice operator [] states that either P or Q may execute. If P performs an event first, then P takes control. Otherwise, Q takes control. In addition to that, PAT supports guarded processes, a guarded process only executes when its guard condition is satisfied. In PAT, a guard process is written as follows,$[cond]P$ where $cond$ is a Boolean formula and P is a process. If $cond$ is true, P executes. It is also important to note that ***assertion*** keyword is used as a query to model the system behaviors. In this research refinement assertion is used, which compares the whole behaviors of a given process with another process, e.g., whether there is a subset relationship.

To model the privacy policies, each privacy policy is defined as a guarded process, in which the condition of the process is the recipient of data. All the privacy processes are combined under general choice operator to create the privacy policy model. In run time each data sharing action or new policy is transformed to a PAT process then a refinement assertion is used to check whether the data sharing action or new policy process complies with the privacy policy model or not.

4. Case Study

A Java program that accepts users privacy policies as an input and transforms each privacy policy to a PAT readable process is implemented. The program then combines all the processes to build privacy policy models. So far, the recipient of each policy as the guard of the process is selected to reduce the number of states generated by model checker. When a data sharing function is called, before the function execution, the data necessary for validation are transformed to PAT readable property. Model-checker executes and if the action is valid, the function continues its execution, else the program informs the user of the violation and asks for the user's decision.

5. Conclusion and Future Works

With the vast amount of information sharing, users need more customized and complex privacy policies. Additional probability and real-time modules of PAT model checker will be used in future work in order to find possibility of information leak and generation of more complex privacy policies.

6. Acknowledgment

This work was supported by National Science Foundation (NSF) award No. 1657774.

References

[1] J. Lu, Z. Huang, and C. Ke, "Verification of behavior-aware privacy requirements in web services composition." *JSW*, vol. 9, no. 4, pp. 944–951, 2014.

[2] E. M. Clarke, O. Grumberg, and D. E. Long, "Model checking and abstraction," *ACM transactions on Programming Languages and Systems (TOPLAS)*, vol. 16, no. 5, pp. 1512–1542, 1994.

[3] C. A. Brodie, C.-M. Karat, and J. Karat, "An empirical study of natural language parsing of privacy policy rules using the sparcle policy workbench," in *Proceedings of the second symposium on Usable privacy and security*. ACM, 2006, pp. 8–19.

[4] J. Sun, Y. Liu, J. S. Dong, and J. Pang, "Pat: Towards flexible verification under fairness." in *CAV*, vol. 9. Springer, 2009, pp. 709–714.