Towards Privacy Policy-Aware Web-Based Systems

Ekaterina Pek
pek@uni-koblenz.de

ADAPT Lab,
Universität Koblenz-Landau,
Koblenz, Germany

1 Problem Description and Motivation

The Web provides different ways to communicate and perform activities that vary from simple to sophisticated: using forums on websites, online shopping, orchestrating Web Services. All of them involve processing data about the user, be it technical (e.g., the version of the user's browser or the IP address) or personal information (name, address, gender, credit card number, etc.). In order to give the user control over their data on the Web, P3P, the Platform for Privacy Preferences, [5] was officially recommended by the World Wide Web Consortium in 2002. P3P is a language and a protocol allowing websites to describe data practices: what data is collected, what for, how long it will be stored and what parts of the data are exposed to other parties. These policies are declarative and non-executable, which leads to the topic of presented work: How to make Web-based systems comply with declared policies? How to make systems policy-aware?

2 Overview of Related Work

2.1 Previous Attempts at P3P Enforcement

Agrawal et al. [6] proposed translating a P3P policy into a set of restrictions in relational database management system (RDBMS) on the level of columns, rows or cells, provided that a user query would contain information about purpose and recipient. This approach requires the support from database producers, since it proposes new language constructs and implementation design for fine-grained access control.

Ashley [7] suggested implementation of an external policy framework that has integration points (Privacy Monitors) between the Privacy Server that contains policies and the application environment. This approach uses Reference Monitors (see below).

IBM developed Tivoli Privacy Manager [1] and related technologies: the Declarative Policy Monitoring [8] and Reference Monitor [12], but the Privacy Manager was withdrawn from marketing in 2009 [2] and corresponding technologies have been retired.

In the work of Hayati and Abadi [10] the authors develop a language-based approach for modeling and verifying aspects of privacy policies. They use the programming language Jif [4], an extension of Java with information-flow types, to show how to prevent leaks of the data from the system and how to implement P3P notions of purposes and retentions. However, this work does not cover all aspects of the P3P language, e.g., base/custom data schemes.

2.2 P3P as an Intermediate Representation

Karjoth et al. [13] proposed the Platform for Enterprise Privacy Practices (E-P3P), which uses P3P to present a coarser-grained privacy policy to the customer, while for internal enforcement a new language is suggested.

In the work of Salim et al. [17], P3P policies are used as an intermediate level between the extended Digital Rights Management model and the user, because P3P preferences are more abstract than a license and it's easier for data owners to specify the purposes for which data is to be collected. In the end, P3P preferences are transformed into MPEG REL (Moving Picture Expert Group Rights Expression Language) licenses that can be enforced by the framework.

2.3 General Solutions to Policy Enforcement

He and Antón [11] propose a framework to bridge the gap between high-level privacy requirements and low-level access control policies by modeling privacy requirements in the role engineering process. The framework provides a basis for enforcing privacy requirements with RBAC (role-based access control) model. The work does not address any high-level privacy requirements language in particular, though, uses P3P elements (e.g., purpose) as an example of standard privacy policy entities.

Mont et al. [15] introduce a notion of "sticky policy" in order to prevent leaking of personal information. The proposed privacy model involves Tracing Authorities as a main point to log and audit the disclosures of confidential data as well as to notify the owner of the data. Such a model requires a request to the Tracing Authority each time when a service wants to transfer the data outside.

Ringelstein and Staab [16] introduce a notion of "sticky logging" in order to collect different kinds of data usage (create, copy, read, update, transfer, delete) in distributed environments. This allows to reconstruct the execution afterwards, which might be useful, if the customer requests the report about data usage. This work does not directly address any kind of compliance of a reconstructed execution with existing privacy policies of a system.

3 Proposed Solution

The relation between the system and the policy can be twofold.

One case is that a P3P policy is created for the existing system by analyzing the behaviour of the system and translating it into the P3P language. This means

that a P3P policy can be seen as a by-product of the system and ideally could be generated from the system.

The second case is that a P3P policy exists before the system. This means that a P3P policy can be seen as a specification of the system or as additional constraints at the modelling phase of the system.

While these scenarios are polar, we hope to bridge them in practical experiments (for more details see Section 4). In the end, we aim to develop language support for describing policies as part of the programming effort: that is, a policy-aware programming language supporting idioms for expressing policy-related constraints. We see this language as a simplified, idealized language or calculus, similar to other language design efforts that use Featherweight or Classic Java.There may be the following language constructs for privacy awareness: annotations of the data model with the privacy-related categories; annotations of persistence actions with duration or access information; annotations of service calls so that sharing of data is classified.

4 Research Method

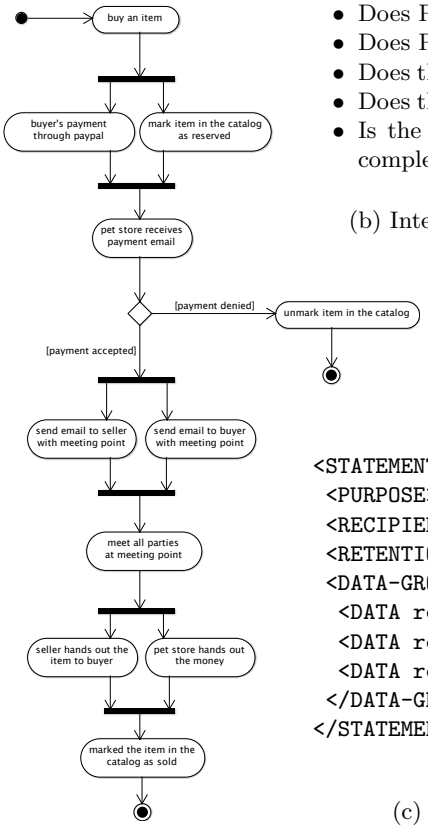
We started our research with an empirical study of P3P policies in the wild [14]. We believe that an empirical study of the language at hand is an important stage of working with the language. While one can start from the specification of the language and try to devise a solution top-down, it can be the case that some combinations of language elements/constructs are seldom or never used, inadequate in practice, or even contradictory [18, 14].

For example, we've found out, that even with low P3P usage – for the seed of 1,450,660 URLs, we were able to download only 4,158 XML files with P3P policies[1]– the coverage of the base data schema proposed in P3P specification is 76%. In other words, the language is used to its full extent and there can be no short-cuts in our effort.

Now that we know the shape of P3P policies, we can start with a realistic and interesting case study to guide the development of a prototype system. This is our plan of attack:

- First, we consider a typical lightweight architecture for a Web-based system: persistence layer, domain-specific logic, presentation layer and, optionally, Web Services. At this point we have decided to dedicate ourselves to Java Pet Store [3], a sample Web application, designed to run on the Java Enterprise Edition 5 platform. While this application has all technical aspects highlighted above, it also suggests non-trivial information flow issues (see Fig. 4). We intend to put some developing efforts into the system in order to make it complete w.r.t. those issues.

[1] Please, note, that this low usage is partially because our approach used one seed of URLs (Google Directory). Cranor et al. [9] found that P3P had been deployed on 10% of the sites returned in the top-20 results of typical searches, and on 21% of the sites returned in the top-20 results of e-commerce searches.



(a) The scenario

- Does PayPal get the buyer's contact information?
- Does PayPal get the seller's contact information?
- Does the seller get the buyer's contact information?
- Does the buyer get the seller's contact information?
- Is the contact information deleted upon purchase completion?

(b) Interesting questions about information flow

```
<STATEMENT>
 <PURPOSE><current/></PURPOSE>
 <RECIPIENT><ours/></RECIPIENT>
 <RETENTION><indefinitely/></RETENTION>
 <DATA-GROUP>
  <DATA ref="#user.name"/>
  <DATA ref="#user.home-info.online.email"/>
  <DATA ref="#user.home-info.postal"/>
 </DATA-GROUP>
</STATEMENT>
```

(c) A P3P policy (excerpt) for the seller

Fig. 4: Privacy concerns in a Web purchase scenario

– Then we write a P3P policy for the system, capturing the system's behaviour in the P3P language. Once we have the policy and the system, which is *unaware* of it, we try to map system's parts to policy's parts – to understand the system in policy terms. To this extent we intend to use some sort of a dynamic, run-time analysis so that to see the flow of customer-related data.
– After that we experiment with different ways to achieve policy-awareness, using such mechanisms as annotations, aspects, assertions, etc. From these experiments we expect insights essential to suggest a development methodology for policy-aware systems.

References

1. IBM Tivoli Privacy Manager Solution Design and Best Practices. IBM Press (2003)

2. IBM Tivoli Privacy Manager info page. http://www-01.ibm.com/software/tivoli/products/privacy-mgr-e-bus/ (Jul 2010)
3. The Java Pet Store 2.0 Reference Application. http://java.sun.com/developer/releases/petstore/ (Sep 2010)
4. Jif home page. http://www.cs.cornell.edu/jif/ (Sep 2010)
5. W3C, the platform for privacy preferences 1.1 (P3P1.1) specification. http://www.w3.org/TR/P3P11/ (Jul 2010)
6. Agrawal, R., Bird, P., Grandison, T., Kiernan, J., Logan, S., Rjaibi, W.: Extending relational database systems to automatically enforce privacy policies. In: ICDE '05: Proceedings of the 21st International Conference on Data Engineering. pp. 1013–1022. IEEE Computer Society (2005)
7. Ashley, P.: Enforcement of a P3P privacy policy. In: Proceedings of the 2nd Australian Information Security Management Conference, Securing the Future. pp. 11–26. School of Computer and Information Science, Edith Cowan University, Western Australia (2004)
8. Bohrer, K., Hada, S., Miller, J., Powers, C., Wu, H.f.: Declarative Privacy Monitoring for Tivoli Privacy Manager. http://www.alphaworks.ibm.com/tech/dpm (Jul 2010)
9. Cranor, L.F., Egelman, S., Sheng, S., McDonald, A.M., Chowdhury, A.: P3P deployment on websites. Electronic Commerce Research and Applications 7(3), 274–293 (2008)
10. Hayati, K., Abadi, M.: Language-based enforcement of privacy policies. In: Proceedings of Privacy Enhancing Technologies Workshop (PET). Springer-Verlag (2004)
11. He, Q., Antón, A.I.: A framework for modeling privacy requirements in role engineering. In: Proceedings of the 9th International Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ'03) (2003)
12. Hill, R.K., Fritz, P.: Reference Monitor for Tivoli Privacy Manager. http://www.alphaworks.ibm.com/tech/refmon (Jul 2010)
13. Karjoth, G., Schunter, M., Waidner, M.: Platform for enterprise privacy practices: privacy-enabled management of customer data. In: PET'02: Proceedings of the 2nd international conference on Privacy enhancing technologies. pp. 69–84. Springer-Verlag (2003)
14. Lämmel, R., Pek, E.: Vivisection of a non-executable, domain-specific language; Understanding (the usage of) the P3P language. In: Proceedings of ICPC 2010. IEEE (2010)
15. Mont, M.C., Pearson, S., Bramhall, P.: Towards accountable management of identity and privacy: Sticky policies and enforceable tracing services. In: DEXA '03: Proceedings of the 14th International Workshop on Database and Expert Systems Applications. p. 377. IEEE Computer Society (2003)
16. Ringelstein, C., Staab, S.: DIALOG: Distributed auditing logs. In: ICWS-2009 - 7th IEEE Int. Conference on Web Services. Los Angeles, CA, USA (2009)
17. Salim, F., Sheppard, N.P., Safavi-Naini, R.: Enforcing P3P policies using a digital rights management system. In: PET'07: Proceedings of the 7th international conference on Privacy enhancing technologies. pp. 200–217. Springer-Verlag (2007)
18. Yu, T., Li, N., Antón, A.I.: A formal semantics for P3P. In: SWS '04: Proceedings of the 2004 workshop on Secure web service. pp. 1–8. ACM (2004)