Privacy Design Patterns and Anti-Patterns

Patterns Misapplied and Unintended Consequences

Nick Doty
UC Berkeley, School of Information
npdoty@ischool.berkeley.edu

Mohit Gupta
Clever, Inc.
mohit.ed@ischool.berkeley.edu

ABSTRACT

One critique of Privacy-by-Design has focused on its lack of concrete guidance for implementation. We have proposed privacy design patterns (drawing from architectural design patterns and object-oriented programming design patterns) as documentation that can be more directly applicable and have established a site to coordinate collaborative development of such patterns. We argue that patterns — which define a problem and a solution within a certain context — can also help us understand and classify many *anti*-patterns as patterns misapplied.

Rather than providing examples of poor or perverse user interface, we examine several design anti-pattern examples, describing: applying a pattern to a different problem; for a different audience; and, with unintended consequences, advantageous and not. With those models, we provide possible directions for how the community should document patterns and anti-patterns to improve future designs.

1. PRIVACY PATTERNS

1.1 Tools for Practical Privacy-by-Design

The phrase "Privacy-by-Design" as it is commonly used around information technologies originated from Ann Cavoukian, Information and Privacy Commissioner for Ontario, Canada, and substantiated as a set of seven principles [4]. The concept has subsequently been popularized by regulators (including significant outreach from Cavoukian's office to build case studies of practice; promotion from the Federal Trade Commission in their 2012 report [6]) and largely welcomed by industry. The phrase is itself ambiguously applied; some, for example, want to contrast privacy enforced through engineering with privacy as protected by law, policy and norms. Critiques of Privacy-by-Design have tended to focus on the step to translate these high-level principles into concrete engineering practice. Rubinstein and Good note the distinction between Fair Information Practices and more concrete software engineering requirements [15]. Mulligan and King note the "gap" between Privacy-by-Design and the HCI design practice of discovering contextual values like privacy [12].

We have proposed privacy design patterns as one of what we hope and expect will be a number of tools to help bridge this gap and translate principles encouraging support of privacy into engineering techniques that do so. Design patterns are abstract solutions to common problems within particular contexts. Historically, design patterns are a construct from architecture and urban planning but are also commonly applied within the software developer community. As a familiar method and one suited to documenting problems and solutions in a variety of contexts, we propose it as a "bottom-up" tool for concrete privacy-by-design.

1.2 History of Design Patterns

Christopher Alexander writes that a "pattern describes a problem that occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice" [2]. Since his initial description of patterns for architecture, design patterns have been described for other disciplines such as software engineering [22], interaction design [20] and education [7].

As a structured method of describing solutions to common problems within a context, patterns can be used to share design practices within a field. While the actual structure of patterns varies across disciplines and authors, most patterns describe the problem-solution pair, the conflicting forces in which the problem expresses itself as an issue, along with the benefits and pitfalls of the solution [1] [3].

The focus on context and repeatability across scenarios motivate a shared language within a discipline, but can more importantly allow for a critique of practices that loose validity in new contexts. Documentation and sharing of software design practices popularized by the seminal Design Patterns book by the Gang of Four [22] has also been important to the discussions about changing common practices as the software design discipline evolves. In fact, some critiques of software engineering patterns have shown how good design patterns can be seen as *pitfalls* in different contexts, as in Norvig's analysis [13].

1.3 Applicability to Privacy

Many existing interaction design, software design and security patterns include problems and solutions associated with privacy. However, privacy and the design of privacy preserving systems is often tangential to the core of these security patterns (see examples from Schumacher in 2003 [18]), as described by Romanosky et al. in presenting three privacy patterns [14]. Collections of components, each effective individually, cannot automatically be considered patterns. Christopher Alexander argued that patterns must represent a configuration of components that resolve a conflict in context [1]. Researchers have shared such privacy patterns that address the security, usability, software engineering aspects of designing for privacy, including Romanosky [14], user in-

terface patterns from Egelman [5] and ongoing work from Hoepman on privacy design strategies as categorizations of privacy design patterns [8].

Describing and sharing experiences of designing for privacy are especially effective as patterns. The design of privacy preserving systems requires special considerations in all stages of the design and development process. Privacy Patterns that span across usability, engineering, security and other considerations can provide sharable descriptions of generative solutions to common design contentions. Since patterns focus on describing the resolutions of contradictory forces in a design context, the pros and cons of a specific solution can be easily debated. Unlike guidelines, regulations or best practices, patterns are descriptive, rather than normative, facilitating discussion and debate and providing education rather than insisting on particular solutions or practices. Design patterns are also easily composable for differing situations and at different levels of granularity, while remaining more actionable compared to privacy design principles such as data minimization or transparency. We believe that privacy patterns are actionable and contestable expressions of privacy design principles in specific contexts.

Development of pattern languages and pattern libraries has historically been a collaborative and ongoing process; in fact, the very first wiki software was created for facilitating development and sharing of patterns on the Portland Pattern Repository in 1995.[1] To that end, we have begun development of a set of privacy design patterns (initially focused on location-based services), available at `http://privacypatterns.org`, and accept collaborative contributions through GitHub.

1.4 Sample Privacy Patterns

Several full privacy patterns (not included here, for space reasons) are available at privacypatterns.org. Particular examples of interest are below, with title and brief descriptions of each problem:

- *Asynchronous notice*:[2] How can a service effectively provide notice to a user who gave permission once but whose information is accessed repeatedly (perhaps even continuously) over a long period of time?
- *Privacy dashboard*:[3] How can a service succinctly and effectively communicate the kind and extent of potentially disparate data that has been collected or aggregated by a service?
- *Location granularity*:[4] Collecting more information than is necessary can harm the user's privacy and increase the risk for the service (in the case of a security breach, for example), but location data may still need to be collected to provide the service.

[1]The PPR and its associated wiki are still online and functional: `http://c2.com/cgi/wiki`.

[2]`http://privacypatterns.org/patterns/Asynchronous-notice`

[3]`http://privacypatterns.org/patterns/Privacy-dashboard`

[4]`http://privacypatterns.org/patterns/Location-granularity`

2. ANTI-PATTERNS

With that background, how do design patterns help us identify and understand *anti*-patterns? Understanding a pattern as a solution to a problem within a context, an anti-pattern may just be a pattern focused on a problem other than the one intended (or the one that the reader has in mind), applied in a different context or use a technique with unintended consequences. While some anti-patterns might be the perverse case — a system specifically designed to frustrate a user — we believe anti-patterns will commonly be patterns applied to different audiences or better suited to different problems. We illustrate with a few example "anti"-patterns: the privacy policy, passwords for delegated authentication and security images for site authentication.

2.1 Privacy Policies

The literature on the inadequacy of privacy policies is well-documented, and, we think, conclusive enough that it does not need to be continued. As an empirical matter, users do not commonly read privacy policies and it would be infeasible to read the privacy policies of all the sites a typical Web user visits, much less the policies of every third-party service, service provider or affiliate company [11]. Furthermore, the presence of a privacy policy link has been shown to give users an incorrect sense of trust in an organization's data management practices (as described by Turow [21] and Hoofnagle & King [9]).

Nevertheless, we consider the privacy policy a privacy design pattern. And, if we consider a different problem, the FTC's move to encourage adoption of Web site privacy policies (subsequently applied, for example, to mobile app stores in California-specific regulation) has been successful. While privacy policies do not commonly succeed in educating the typical end user about the actual data practices, they can often be successful in other areas, including:

- enabling power users or particularly interested users to learn about specific data practices
- facilitating self-regulation and expert watchdogs to uncover and document unsavory behaviors
- providing a regulatory "hook" for enforcement through consumer protection laws (like Section V)
- encouraging organizations to go through a self-analysis of data-handling practices

For technical designers who want to solve the problem, "how can I let a particularly interested user know the details of my data handling practices without overwhelming common users with information?" a privacy policy is an ideal pattern to apply.

Although not considered in the basic design pattern concept, there may also be ancillary benefits to the use of certain patterns that aren't directly related to the application of an abstract solution to a problem in a particular context. Creating regulatory liability (that if our organization makes a material assertion the FTC can bring an enforcement action if we violate it) and serving as a forcing function for self-reflection and internal documentation of data management may not be the intentions of a designer applying the privacy policy design pattern. However, those benefits to increased end-user privacy may meaningfully explain why regulators encourage the adoption of privacy patterns despite their lack of success in educating the typical user.

2.2 Passwords for Delegated Authentication

It is now not uncommon to hear the phrase "password anti-pattern" in critique of computer security's reliance on secret passphrases. We trace its use back at least as far as 2007,[5] in describing the considered-harmful practice of asking for a user's email or social networking password on an unrelated site in order to authenticate the user or provide remote access to some piece of data (a contact list for use in finding friends, for example). This specific example might be more precisely defined as "third-party passwords for delegated authentication" with the alternative being protocols like OAuth[6] that enable delegated authorization.

While previously we saw ancillary benefits of the privacy policy pattern (even when it was not applied to the suitable problem), in this case we see an ancillary harm. For an online service that wants to provide friend-finding, Third-Party Passwords for Delegated Authentication might successfully and reasonably easily solve the problem, with little security risk.[7] However, the pattern itself persuades users to provide their password for one service to some unrelated service. As commenter Simon Willison[8] puts it, this pattern "teaches users to be phished". We find ourselves in a collective action problem: the actions of any particular third-party service can erode the security of the Web as a whole, by making commonplace a practice externally indistinguishable from phishing attacks, without suffering individual consequences.

Supporting OAuth or similarly structured protocols may help address this problem. Some services have prohibited (in their terms, or with technical measures) third-party access with passwords (for example, Fire Eagle and Twitter). We can also document the consequence, as we intend to with privacy patterns, because in some cases this will depend on the choices of individual technical designers. As Keith put it in the aforementioned blog post, the use of this anti-pattern is a *moral* question for developers.

Identifying a pattern and its misapplication as an anti-pattern does not itself answer the empirical question of users' impressions of trust in a particular service. Does it affect a user's trust of a particular site when they're asked for a password for another service? However, understanding the pattern does raise an additional research question, which we suggest is more germane: does it affect a user's trust of the Web as a whole?

2.3 Security Images for Site Authentication

One class of anti-patterns are those features intended to provide additional security without actually improving security outcomes. To give one such example, in 2007 researchers conducted a usability and security study of a set of "website authentication indicators" — features where financial institutions presented a user-chosen image during the login process so that the user could be assured they were logging into their bank and not a phishing site. In Schechter et al.'s study [16], however, almost all users provided their password even when the authentication image was missing, or replaced with an error message. Because the site authentication indicator only improves security when users confirm the absence of the indicator as a reason not to provide their password, the authors conclude that the features are ineffective.[9] Nevertheless, site authentication indicators continue to see use.

We might indicate this practice as an anti-pattern because it provides a false sense of security, encouraging users to believe at some point that their passwords are additionally protected from phishing or other site forgery while actually not providing that aid. However, in understanding why site authentication indicators are still used might help us understand this pattern differently. In a news report following publication of the usability study, a bank representative indicated that the site indicator improved security as one of many layered defenses and an author of the study speculated that "[instilling] trust and confidence and good feelings" might have been the motivation for adoption [19]. Related, the study authors note that US financial regulations requiring additional security measures might explain the rapid implementation of this particular technology. In understanding patterns as directed at a particular problem, a particular audience and a particular context, these apparent anti-patterns might be cases where the problem being effectively solved is a lack of trust and confidence (rather than phishing or site forgery) or the audience is policymakers or regulators rather than customers.[10]

Similar arguments have been made about cases of "security theater" as described by Bruce Schneier, where certain very visible practices are implemented without meaningfully decreasing risks but providing comfort that actions are being taken to decrease risk. Schneier explicitly notes some positive applications of what he has more commonly criticized as security theater — security measures that help people feel more comfortable, especially when a risk might be very rare but very visible (and thus likely to be overestimated) [17]. If supporting the multi-faceted concept of privacy includes providing a sense of trust, we may conclude that certain privacy "anti-patterns" can be classified as privacy patterns aimed at solving that lack of trust.

3. CONCLUSIONS

These classes of anti-patterns give us an indication of how anti-patterns get applied by designers in real-world implementations and potentially how to avoid them. In the misapplication of privacy policies, for example, a designer may have correctly identified that privacy policies are used to implement the principle of transparency, but did not more precisely tailor use of the privacy policy to the problem that it addresses. In each case, while we observe patterns in individual implementations, patterns provide insight into the technical design process.

These examples can also help guide the process of docu-

[5] `http://adactio.com/journal/1357`

[6] `http://oauth.net`

[7] Doesn't sharing a password in plain text have inherent security risk? Perhaps, but a site could implement handling passwords for delegated authentication in relatively safe ways that don't include storing the password. Those hypothetical implementations would still have the collective privacy concern we document here.

[8] `http://lanyrd.com/profile/simonw`

[9] In our understanding, a true man-in-the-middle attacker could also fetch the image from the bank site and present it to the user in the second phase of login.

[10] Academic studies (Leon, Ur, et al. [10], for example) have critiqued the usability of opt-out mechanisms for online behavioral advertising; one possible explanation might be that the audience for such tools are the regulatory agencies that might consider intervention where consumer opt-outs are not available.

menting privacy patterns for contributors to that project. In the privacy policy anti-pattern example, highlighting which problems a pattern can be applied to successfully and which it may not help can provide education to avoid misapplied patterns. We have also identified ancillary effects, potentially unintended consequences both positive and negative towards the goal of robust online privacy, present in patterns. By identifying, naming and documenting patterns, we also have an opportunity to describe the risks and forces that appear in applying a particular technique.

Finally, understanding privacy design patterns can let us distinguish among different "trustbuster" cases — some designs and architectures inhibit trust or undermine privacy intentionally or perversely, but in some cases patterns are misapplied or have consequences for privacy not directly intended. We believe distinguishing between these cases may be useful guides for policymakers who wish to address privacy anti-patterns; solutions will require a mix of both enforcement and education.

4. ACKNOWLEDGMENTS

Thanks to several audiences for providing feedback on the Privacy Patterns project, including: students and faculty of the UC Berkeley School of Information (in particular, Professor Deirdre K. Mulligan), WhereCamp, South-by-Southwest Interactive, the Asia Pacific Privacy Authorities, the Institute for Information Infrastructure Protection, reviewers for the Symposium on Usable Privacy and Security; and colleagues at Location Labs, Lookout and Foursquare.

This material is based in part upon work supported by the U.S. Department of Homeland Security under grant award #2006-CS-001-000001 and the National Institute of Standards and Technology, under grant award #60NANB1D0127, under the auspices of the Institute for Information Infrastructure Protection (I3P) research program. The I3P is managed by Dartmouth College. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the U.S. Department of Homeland Security, the I3P, or Dartmouth College.

The Berkeley Center for Law and Technology supported this research project in part in conjunction with a research gift from Nokia.

5. REFERENCES

[1] C. Alexander. *The timeless way of building*, volume 1. New York: Oxford University Press, 1979.

[2] C. Alexander, S. Ishikawa, and M. Silverstein. *A Pattern Language: Towns, Buildings, Constructions*. Oxford University Press, 1977. URL: http://books.google.com/books?id=hwAHmktpk5IC.

[3] F. Buschmann, K. Henney, and D. C. Schmidt. *Pattern-oriented software architecture, Vol. 5: On Patterns and Pattern Languages*. Wiley, Chichester, England; Hoboken, N.J., 2007. URL: http://www.myilibrary.com?id=85603.

[4] A. Cavoukian. Privacy by Design. Technical report, Information & Privacy Commissioner, Ontario, Canada, 2009. URL: http://www.privacybydesign.ca/content/uploads/2009/01/privacybydesign.pdf.

[5] S. Egelman. *Trust me: design patterns for constructing trustworthy trust indicators*. PhD thesis, 2009. URL: http://books.google.com/books?isbn=1109149131.

[6] FTC. Protecting Consumer Privacy in an Era of Rapid Change Recommendations for Businesses and Policymakers. Technical Report March, Federal Trade Commission, 2012. URL: http://ftc.gov/os/2012/03/120326privacyreport.pdf.

[7] P. Goodyear. Patterns, pattern languages and educational design. In *Beyond the comfort zone: Proceedings of the 21st ...*, pages 339–347, 2004. URL: http://www.ascilite.org.au/conferences/perth04/procs/pdf/goodyear.pdf.

[8] J.-H. Hoepman. Privacy Design Strategies. Technical report, Oct. 2012. URL: http://arxiv.org/abs/1210.6621.

[9] C. J. Hoofnagle and J. King. What Californians Understand about Privacy Online. Technical report, Social Science Research Network, Rochester, NY, Sept. 2008. URL: http://papers.ssrn.com/abstract=1262130.

[10] P. Leon, B. Ur, R. Shay, Y. Wang, R. Balebako, and L. Cranor. Why Johnny can't opt out: a usability evaluation of tools to limit online behavioral advertising. CHI '12, pages 589–598, New York, NY, USA, 2012. ACM. URL: http://doi.acm.org/10.1145/2207676.2207759, doi:10.1145/2207676.2207759.

[11] A. M. McDonald and L. F. Cranor. The Cost of Reading Privacy Policies. *I/S: A Journal of Law and Policy for the Information Society*, 4:543, 2008. URL: http://heinonline.org/HOL/Page?handle=hein.journals/isjlpsoc4&id=563&div=&collection=journals.

[12] D. K. Mulligan and J. King. Bridging the Gap between Privacy and Design. *University of Pennsylvania Journal of Constitutional Law*, 14(4), Apr. 2012. URL: http://papers.ssrn.com/sol3/papers.cfm?abstract_id=2070401.

[13] P. Norvig. Design Patterns in Dynamic Programming, Mar. 1998. URL: http://norvig.com/design-patterns/design-patterns.pdf.

[14] S. Romanosky, A. Acquisti, J. Hong, L. Cranor, and B. Friedman. Privacy patterns for online interactions. In *PLoP 06 Proceedings of the 2006 conference on Pattern languages of programs*, 2006. URL: http://portal.acm.org/citation.cfm?id=1415472.1415486.

[15] I. Rubinstein and N. Good. Privacy by Design: A Counterfactual Analysis of Google and Facebook Privacy Incidents. Technical report, NYU School of Law, Public Law Research Paper No. 12-43, Rochester, NY, Aug. 2012. URL: http://papers.ssrn.com/abstract=2128146.

[16] S. Schechter, R. Dhamija, A. Ozment, and I. Fischer. The Emperor's New Security Indicators: An evaluation of website authentication and the effect of role playing on usability studies. In *Symposium on Usable Privacy and Security*, pages 51–65, 2007. URL: http://ieeexplore.ieee.org/ielx5/4223200/4223201/04223213.pdf?tp=&arnumber=4223213&isnumber=4223201, doi:10.1109/SP.2007.35.

[17] B. Schneier. In Praise of Security Theater. URL: http://www.wired.com/politics/security/commentary/securitymatters/2007/01/72561.

[18] M. Schumacher. Security Patterns and Security Standards - With Selected Security Patterns for Anonymity and Privacy. In *European Conference on Pattern Languages of Programs*, 2003. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.9.4513.

[19] B. Stone. Study Finds Web Antifraud Measure Ineffective, Feb. 2007. URL: http://www.nytimes.com/2007/02/05/technology/05secure.html.

[20] J. Tidwell. Common Ground: A Pattern Language for Human-Computer Interface Design, 1999. URL: http://www.mit.edu/~jtidwell/common_ground.html.

[21] J. Turow. Americans & online privacy: The system is broken. Technical report, Annenberg Public Policy Center, University of Pennsylvania, 2003. URL: http://www.annenbergpublicpolicycenter.org/Downloads/Information_And_Society/20030701_America_and_Online_Privacy/20030701_online_privacy_report.pdf.

[22] J. Vlissides, R. Helm, R. Johnson, and E. Gamma. *Design patterns: Elements of reusable object-oriented software*, volume 49. Reading: Addison-Wesley, 1995.