Compliance Traceability: Privacy Policies as Software Development Artifacts

Sebastian Zimmeck[1], Peter Story[2], Rafael Goldstein[1], David Baraka[1], Shaoyan Li[2], Yuanyuan Feng[2], and Norman Sadeh[2]

[1] Department of Mathematics and Computer Science, Wesleyan University
{szimmeck,rgoldstein01,dbaraka}@wesleyan.edu
[2] School of Computer Science, Carnegie Mellon University
{pstory,shaoyanl,yuanyua2}@andrew.cmu.edu, sadeh@cs.cmu.edu

Abstract. The increasing importance of data privacy is met by legislators around the world with new privacy laws. However, it was shown that many developers struggle to understand the privacy implications of their code and the law they have to adhere to. Commercially available privacy policy generators are intended to help developers to properly disclose their privacy practices by guiding them through a questionnaire and generating a privacy policy based on the answers to the presented questions. However, state-of-the-art generators are inherently limited by the error-prone nature of the questionnaire-based approach. They are necessarily reliant on developers' ability to answer questions on their programs' functionality accurately as well as their timeliness in updating their answers in order to reflect software changes.
We propose to mitigate the potential for compliance issues by leveraging code analysis for purposes of generating privacy policies. We design and implement a policy generator for iOS apps written in Swift that uses three resources: (1) templates with standard language that every policy must include to be compliant with legal provisions, for example, the General Data Protection Regulation (GDPR), (2) policy snippets generated from an app's source code analysis, and (3) developer input via a wizard-based user interface. We think that privacy transparency can be increased and compliance issues reduced by establishing policy generation as a native extension of the software development process. Ultimately, the integration of policy generation into the software development process enables the traceability of compliance requirements originating from new privacy laws and the evolving regulatory landscape.

Keywords: privacy · compliance · privacy policies · static analysis · mobile applications · software development · iOS

Our approach is to integrate privacy analysis and policy generation in three stages: template provisioning, code analysis, and wizard fine tuning. Figure 1 shows an overview of our approach as well as a sample analysis result from our implementation.

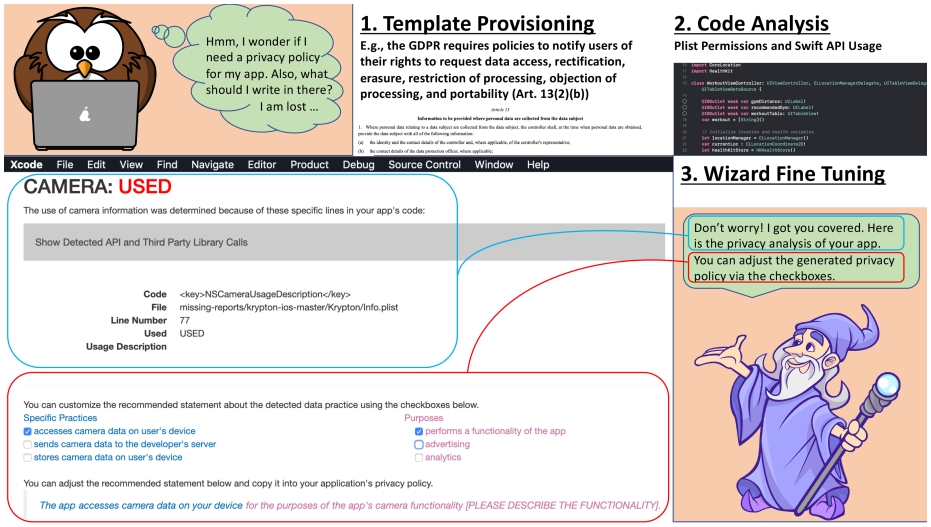

Fig. 1. The three phases of our policy generation approach and a partial result output file from our implementation. The automatic code analysis revealed that the analyzed app is using the camera for its main functionality, a finding that can be adjusted via the wizard's checkboxes. A generated privacy policy statement is shown in the last line.

Various privacy laws, notably the GDPR, require mandatory disclosures that are the same for any software. For example, developers must disclose that users have the right to access the data they have stored on them (Art. 13(2)(b), 14(2)(c)). Those types of disclosures can be included in a template.

Generating policies from source code holds the advantage that it does not require any additional knowledge or effort beyond writing the respective code. It brings the policy creation task squarely into the domain of software development and supports a mental model developers are already familiar with. In addition, it prevents compliance issues, that is, disconnects between code and policy. We implemented our generator for iOS apps that are written in Swift. It can be used as a standalone Python application, but we also integrated it via a build script into Xcode. If an app wants to make a privacy-sensitive call, the required permission (e.g., for location access) must be included in its Info.plist. The majority of privacy-sensitive APIs are comprised of imports, constructors, and functions that are checked.

After the initial provisioning of the template and the analysis of the app's source code, the developer is presented with the generated policy whose contents can be adjusted via a wizard.