Exploring Societal Computing based on the Example of Privacy

Swapneel Sheth

Submitted in partial fulfillment of the
requirements for the degree
of Doctor of Philosophy
in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2014

ABSTRACT

Exploring Societal Computing based on the Example of Privacy

Swapneel Sheth

Data privacy when using online systems like Facebook and Amazon has become an increasingly popular topic in the last few years. This thesis will consist of the following four projects that aim to address the issues of privacy and software engineering.

First, only a little is known about how users and developers perceive privacy and which concrete measures would mitigate their privacy concerns. To investigate privacy requirements, we conducted an online survey with closed and open questions and collected 408 valid responses. Our results show that users often reduce privacy to security, with data sharing and data breaches being their biggest concerns. Users are more concerned about the content of their documents and their personal data such as location than about their interaction data. Unlike users, developers clearly prefer technical measures like data anonymization and think that privacy laws and policies are less effective. We also observed interesting differences between people from different geographies. For example, people from Europe are more concerned about data breaches than people from North America. People from Asia/Pacific and Europe believe that content and metadata are more critical for privacy than people from North America. Our results contribute to developing a user-driven privacy framework that is based on empirical evidence in addition to the legal, technical, and commercial perspectives.

Second, a related challenge to above, is to make privacy more understandable in complex systems that may have a variety of user interface options, which may change often. As social network platforms have evolved, the ability for users to control how and with whom information is being shared introduces challenges concerning the configuration and comprehension of privacy settings. To address these concerns, our crowd sourced approach simplifies the understanding of privacy settings by using data collected from 512 users over a 17 month period to generate visualizations that allow users to compare their personal settings to an arbitrary subset of individuals of their

choosing. To validate our approach we conducted an online survey with closed and open questions and collected 59 valid responses after which we conducted follow-up interviews with 10 respondents. Our results showed that 70% of respondents found visualizations using crowd sourced data useful for understanding privacy settings, and 80% preferred a crowd sourced tool for configuring their privacy settings over current privacy controls.

Third, as software evolves over time, this might introduce bugs that breach users' privacy. Further, there might be system-wide policy changes that could change users' settings to be more or less private than before. We present a novel technique that can be used by end-users for detecting changes in privacy, i.e., regression testing for privacy. Using a social approach for detecting privacy bugs, we present two prototype tools. Our evaluation shows the feasibility and utility of our approach for detecting privacy bugs. We highlight two interesting case studies on the bugs that were discovered using our tools. To the best of our knowledge, this is the first technique that leverages regression testing for detecting privacy bugs from an end-user perspective.

Fourth, approaches to addressing these privacy concerns typically require substantial extra computational resources, which might be beneficial where privacy is concerned, but may have significant negative impact with respect to Green Computing and sustainability, another major societal concern. Spending more computation time results in spending more energy and other resources that make the software system less sustainable. Ideally, what we would like are techniques for designing software systems that address these privacy concerns but which are also sustainable — systems where privacy could be achieved "for free," i.e., without having to spend extra computational effort. We describe how privacy can indeed be achieved for free — an accidental and beneficial side effect of doing some existing computation — in web applications and online systems that have access to user data. We show the feasibility, sustainability, and utility of our approach and what types of privacy threats it can mitigate.

Finally, we generalize the problem of privacy and its tradeoffs. As Social Computing has increasingly captivated the general public, it has become a popular research area for computer scientists. Social Computing research focuses on online social behavior and using artifacts derived from it for providing recommendations and other useful community knowledge. Unfortunately, some of that behavior and knowledge incur societal costs, particularly with regards to Privacy, which is viewed quite differently by different populations as well as regulated differently in different locales.

But clever technical solutions to those challenges may impose additional societal costs, e.g., by consuming substantial resources at odds with Green Computing, another major area of societal concern. We propose a new crosscutting research area, *Societal Computing*, that focuses on the technical tradeoffs among computational models and application domains that raise significant societal issues. We highlight some of the relevant research topics and open problems that we foresee in Societal Computing. We feel that these topics, and Societal Computing in general, need to gain prominence as they will provide useful avenues of research leading to increasing benefits for society as a whole.

Table of Contents

List of Figures

List of Tables

Acknowledgments

First, and foremost, I would like to thank my advisor, Gail Kaiser, for the invaluable support and advice over the years.

I would like to thank, in particular, Chris Murphy, Adam Cannon, and Walid Maalej for mentoring me about different aspects of Ph.D. life. I would also like to thank other members of my committee — Tal Malkin, Augustin Chaintreau, and Steve Bellovin.

I would like to thank past and present members of the Programming Systems Lab: Jonathan Bell, Nipun Arora, Leon Wu, Mike Su, Riley Spahn, and Lindsay Neubauer. A big thanks to my “old friends”: Vinaya Candes, Aaron Fernandes, Vaibhav Saharan, Shweta Savoor, Amortya Ray, Siddharth Kulkarni, and Siddarth Adukia; my“new(er) Columbia friends”: Pablo Barrio, Marcin Szczodrak, Chris Riederer, Arthi Ramachandran, and Nicolas Dedual; and my “conference buddies”: Dennis Pagano, Leif Singer, and Chris Lewis. Thanks for all the late nights, existential conversations, and fun memories.

I’ve taught five classes at Columbia and I’ve had a great time doing it — so a big shout out and thanks to the students and TAs in my classes. You know who you are! Thanks to all the undergrad and masters students who I’ve had the pleasure to work with during my Ph.D.

Finally, last, but not the least, I would like to thank my parents for all the support and being there when I needed them.

Alla vita.. cosi' strana.. cosi' bella

Chapter 1

Introduction

1.1 Privacy

Privacy in computing systems has become increasingly important in the last few years. There have been many different definitions of privacy over time as privacy has always been a concern historically. In earlier times, however, privacy usually referred to physical privacy. Privacy and its definitions have many roots in the laws and the legal system in the United States and in Europe. One of the earliest definitions was the "right to be left alone" as described by Warren and Brandeis [161]. More recently, Westin defines the four states of privacy: solitude, intimacy, anonymity, and reserve [162]. This definition focuses on the notion of "social distance". Solove broadens the definition of privacy and claims that "privacy is an umbrella term, referring to a wide and disparate group of related things". The author proposes a taxonomy of privacy in the context of harmful activities such as information collection, information processing, information dissemination, and invasion [138]. According to the Merriam-Webster dictionary, privacy is the "freedom from unauthorized intrusion".

The advent of a large number of online systems that collect and analyze user data has brought this into the forefront recently. Westin [162] describes this in a remarkably prescient manner: "The computer-born revolution in human capacity to process data is obviously an enormous boon to mankind. [. . .] At the same time, however, four developments [information gathering and record keeping, the advent of the digital computer, accelerated data sharing among those who collect information, automatic data processing] in the data processing field are beginning to have profound implications for our traditional patterns of privacy."

He goes on to claim that: "Unless the issue of privacy is in the forefront of the planning and administration of such future computer systems, however, the possibilities of data surveillance over the individual in 1984 could be chilling. [...] There would be few areas left in which anyone could move about in the anonymity of personal privacy and few transactions that would not be fully documented for government examination."

1.2 Privacy and Software Engineering

Recent incidents like the leaks of Edward Snowden and NSA Prism show the accuracy of his predictions. In this thesis, we are interested specifically in data privacy. Other notions of privacy such as physical privacy are beyond the scope of this thesis. There has been a lot of work on privacy in different domains such as databases, theory, and economics. The relevant related work in discussed in the following chapters. There has, however, been very limited work on privacy in the software engineering so far.

Thus, this thesis focuses on implications of privacy on various aspects of software engineering. In particular, the typical first step in software engineering is to gather requirements. Most of the work on privacy so far assumes that privacy is well-specified and important. However, there is very little evidence about what exactly are the user concerns, priorities, and trade-offs, and how users think these concerns can be mitigated. In particular, in the software engineering community, there have been no systematic studies to find out what privacy requirements are and how these requirements should be addressed by developers. This is the focus of Chapter 2. We conducted a large online survey to gather privacy requirements from people with and without software development experience and people from different parts of the world. This project illustrates and quantifies general trends and differences in privacy expectations between these groups.

One of the results from this project tells us that end-users do not need complex technical approaches like anonymization for mitigating privacy concerns. They are equally satisfied with more transparency and details about how their data is being used and who can access it. We used this result to build a novel tool that crowdsources privacy settings and shows how a certain user's settings compare to a trusted set of friends. This gives users more insights on what their settings are and, implicitly, what they should be. We describe this project in Chapter 3.

An important step in software engineering is testing. There has been a lot of work and advances in software testing recently. There has, however, been very limited work on doing software testing for privacy. The crowdsourcing approach can be easily extended to help detect privacy bugs in software systems. Further, the novelty of this approach is that it doesn't require access to the source code and can thus be done by end-users. We describe this project in Chapter 4.

Next, we explore privacy and its tradeoffs with green computing. There has been recent work on privacy that shows that privacy concerns can result in a lot of system overhead. We show that, at least in the types of systems under investigation (recommender systems with access to user data), it is possible to get (differential) privacy without any extra computational overhead. We describe this in Chapter 5.

1.3 Privacy and Societal Computing

These previous chapters show the privacy cannot exist in isolation as far as software systems are concerned. There are many tradeoffs that need to be taken into account when building complex software systems. Finally, in chapter 6, we generalize the problem of privacy and these tradeoffs. We propose and define "**Societal Computing**," a new research area for computer scientists in general and software engineering and programming language communities (SE/PL, hereafter) in particular, concerned with the impact of computational tradeoffs on societal issues. Societal Computing research will focus on aspects of computer science that address significant issues and concerns facing the society as a whole such as Privacy, Climate Change, Green Computing, Sustainability, and Cultural Differences. In particular, Societal Computing research will focus on the research challenges that arise due to the tradeoffs among these areas. This thesis has largely focused on privacy. But in this chapter, we describe how the results and artifacts can be applied and reused for exploring the tradeoffs in other areas of Societal Computing.

Chapter 2

Us and Them — A Study of Privacy Requirements Across North America, Asia, and Europe

2.1 Introduction

As systems that collect and use personal data, such as Facebook and Amazon, become more pervasive in our daily lives, users are starting to worry about their privacy. There has been a lot of media coverage about data privacy. One of the earliest articles in the New York Times reported how it was possible to break the anonymity of AOL's search engine's users [14]. A more recent article mentions privacy concerns about Google Glass [101]. Both technical and, especially, non-technical users are finding it increasingly hard to navigate this privacy minefield [67]. This is further exacerbated by well-known systems periodically making changes that breach privacy and not allowing users to opt out a-priori [56].

There is a large body of research on privacy in various research communities. This ranges from data anonymization techniques in different domains [33, 88, 121, 145] to novel approaches to make privacy settings more understandable [55, 118]. Recent studies have shown that there is a discrepancy between users' intentions and reality for privacy settings [91, 94]. The assumption behind most of this work is that privacy is well-specified and important. However, there is very little evidence about what exactly are the user concerns, priorities, and trade-offs, and how users think these concerns can be mitigated. In particular, in the software engineering community, there have been no systematic studies to find out what privacy requirements are and how these requirements should be addressed by developers.

This research aims to understand the privacy expectations and needs for modern software systems. To this end, we conducted an online survey. We received 595 responses and selected 408 of them as valid. The responses represented diverse populations including developers and users, and people from North America, Europe, and Asia. The results of our study show that the biggest privacy concerns are data sharing and data breaches. However, there is a disagreement on the best approach to address these concerns. With respect to types of data that are critical for privacy, respondents are least concerned about metadata and interaction data and most concerned about their personal data and the content of documents. Most respondents are not willing to accept less privacy in exchange for fewer advertisements and financial incentives such as discounts on purchases.

The main contribution of this project is threefold. First, it illustrates and quantifies the general trends on how users understand privacy and on how they assess different privacy concerns and measures to address them. Second, the project identifies differences in privacy expectations between various groups: developers versus users and people from different geographic regions. Finally, the project gives insights into how software developers and managers can identify, analyze, and address privacy concerns of their users – building a first step towards a software engineering privacy framework.

Our analysis for geographic regions, for example, shows that there is a significant difference between respondents from North America, Europe, and Asia/Pacific. People from Europe and Asia/Pacific rate different types of data such as metadata, content, and interaction data being a lot *more critical* for privacy than respondents from North America. People from Europe are a lot more concerned about data breaches than data sharing whereas people from North America are equally concerned about the two. Similarly, our analysis for developers versus users shows a marked difference between the two groups. For example, developers believe that privacy laws and policies are *less* effective for reducing privacy concerns than data anonymization.

The rest of the chapter is organized as follows. Section 2.2 describes the design of our study. Sections 2.3, 2.4, and 2.5 highlight its key results. Section 2.6 discusses the implications of the results and their limitations. Finally, Section 2.7 describes related work and Section 2.8 concludes the chapter.

2.2 Study Design

We describe the research questions, methods, and respondents of our study.

2.2.1 Research Questions

There have been many different definitions of privacy over time. One of the earliest definitions was the "right to be left alone" as described by Warren and Brandeis [161]. Solove claims that "privacy is an umbrella term, referring to a wide and disparate group of related things". The author proposes a taxonomy of privacy in the context of harmful activities such as information collection, information processing, information dissemination, and invasion [138]. According to the Merriam-Webster dictionary, privacy is the "freedom from unauthorized intrusion". We are interested specifically in data privacy and other notions of privacy such as physical privacy are beyond the scope of our work.

The goal of this study is to gather and analyze privacy requirements for modern software systems. In particular, we want to study the perception of different groups of people on privacy. We focused on the following research questions:

- **RQ 1**: What are developers' and users' perceptions of privacy? What aspects of privacy are more important and what are the best measures to address them? (Section 2.3)
- **RQ 2**: Does software development experience have any impact on privacy requirements? (Section 2.4)
- **RQ 3**: Does geography have any impact on privacy requirements? (Section 2.5)

By perception, we mean the *subjective* understanding and assessment of privacy aspects. Since privacy is a very broad term, we are interested in specific *aspects*, in particular, types of concerns, measures to mitigate these concerns, types of data that are critical to privacy, and whether people would give up privacy. We think these aspects are most related to software and requirements engineering concerns.

2.2.2 Research Method

We designed an online survey with 16 questions, which took 5–10 minutes to answer. Out of the 16 questions, 14 were closed and respondents had to choose an answer from a list of options. The survey also had two open-ended questions. This helped us get qualitative insights about privacy and gave an opportunity for respondents to report aspects that were not already included in the closed questions.

We chose a survey instead of observations or interviews for the following reasons. First, surveys are scalable and allow to get a large number and broad cross-section of responses. Second, we were interested in the subjective opinion of people and this can be different from real behavior. Third, the closed questions were purely quantitative and allowed us to analyze general trends and correlations. We did not aim for a representative report of the opinions. This would have been possible only through a well-defined population and random representative sampling. Instead, we were interested in the priority trends and inter-relations, which can be analyzed through a cross-tabulation of the survey answers.

We used semantic scales for the closed questions, allowing for the measurement of subjective assessments while giving respondents some flexibility of the interpretation [127]. For example, one question was: "Would users be willing to use your system if they are worried about privacy issues?" and the answer options were: "Definitely yes — Users don't care about privacy", "Probably yes", "Unsure", "Probably not", and "Definitely not — if there are privacy concerns, users will not use this system". To reduce the complexity of matrix questions (which include multiple answer options) we used a 3-point scale consisting of "Yes", "No", and "Uncertain". When we observed in the dry runs that higher discriminative powers were needed, we used a 5-point scale [75].

Respondents could choose to fill out our survey in two languages: English or German. For each language, there were two slightly different versions based on whether the respondents had experience in software development or not. The difference in the versions was only in the phrasing of the questions in order to reduce confusion. For example, developers were asked: "Would users be willing to use your system if they are worried about privacy issues?" whereas users were asked: "Would you be willing to use the system if you are worried about privacy issues?"

To increase the *reliability* of the study [127], we took the following measures:

- Pilot Testing: We conducted pilot testing in four iterations with a total of ten users that focused on improving the timing and understandability of the questions. We wanted to reduce ambiguity about the questions and answers and ensure that none of the semantics were lost in translation. We used the feedback from pilot testing to improve the phrasing and the order of questions for the English and German versions.
- Random order of answers: The answer options for the closed questions were *randomly* ordered. This ensures that answer order does not influence the response [151].
- Validation questions: To ensure that respondents did not fill out the answers arbitrarily, we included two validation questions [8]. For example, one of the validation questions was: “What is the sum of 2 and 5?” Respondents who did not answer these correctly were not included in the final set of valid responses.
- Post sampling: We monitored the number of respondents from each category of interest: developers, users, and geographic location. We conducted post-sampling and stratification to ensure that we got sufficient responses for each category and that the ratio of developers to users for each geographic location was roughly similar. For categories that did not have sufficient respondents, we targeted those populations by posting the survey in specific channels. We stopped data collection when we had a broad spectrum of respondents and sufficient representation in all the categories.

Finally, to corroborate our results, we conducted a number of statistical tests. In particular, we used the Z-test for equality of proportions [141] and Welch’s Two Sample t-test to check if our results are statistically significant.

2.2.3 Survey Respondents

We did not have any restrictions on who could fill out the survey. We wanted, in particular, people with and without software development experience and people from different parts of the world. We distributed our survey through a variety of channels including various mailing lists, social networks like Facebook and Twitter, personal contacts, and colleagues. We circulated the survey across companies with which we are collaborating. We also asked specific people with many contacts (e.g.,

Table 2.1: Summary of study respondents based on location and software development experience

	Developers	Users
North America	85	44
Europe	116	65
Asia	61	30
South America	3	2
Africa	2	0

with many followers on Twitter) to forward the survey. As an incentive, two iPads were raffled among the respondents.

In total, 595 respondents filled out our survey between November 2012 and September 2013. Filtering out the incomplete and invalid responses resulted in 408 valid responses (68.6%). Table 2.1 shows the respondents based on location and software development experience. The four versions of the survey along with raw data and summary information are available on our website[1]. Among the respondents, 267 have software development experience and 141 do not. For respondents with development experience, 28 have less than one year of experience, 129 have 1-5 years, 57 have 5-10 years, and 53 have more than ten years of experience. 129 respondents live in North America, 181 in Europe, and 91 in Asia/Pacific. 166 are affiliated with industry or public sector, 182 are in academia and research, and 56 are students.

2.3 Privacy Perceptions

We asked respondents: “How important is the privacy issue in online systems?” They answered using a 5-point semantic scale ranging from “Very important” to “Least important”. Two thirds of the respondents chose “Very Important”, 25.3% chose “Important”, and the remaining three options (“Average”, “Less Important”, “Least Important”) combined were chosen by a total of 8.1% of the respondents.

[1] http://mobis.informatik.uni-hamburg.de/privacy-requirements/

The *location* of the data storage was a key concern for the respondents. We asked respondents whether privacy concerns depend on the location of where the data is stored and provided a 5-point semantic scale with options: "Yes", "Maybe yes", "Unsure", "Maybe not", and "No". 57.7% of the respondents chose "Yes", 28.6% chose "Maybe yes", while only 13.7% of the respondents chose the remaining three options.

On the other hand, there was disagreement about whether users would be *willing* to use such systems if there were privacy concerns. The answer options were: "Definitely yes — Users don't care about privacy", "Probably yes", "Unsure", "Probably not", and "Definitely not — if there are privacy concerns, users will not use this system". 20.8% of the respondents choose "Unsure", while 34.8% and 29.4% chose "Probably yes" and "Probably not" respectively.

2.3.1 Factors that Increase and Reduce Privacy Concerns

We asked respondents if the following factors would *increase* privacy concerns:

- Data Aggregation: The system discovers additional information about the user by aggregating data over a long period of time.
- Data Distortion: The system might misrepresent the data or user intent.
- Data Sharing: The collected data might be given to third parties for purposes like advertising.
- Data Breaches: Malicious users might get access to sensitive data about other users.

For each concern, the respondents could answer using a 3-point semantic scale with the options: "Yes", "Uncertain", and "No". We also asked respondents if the following would help to *reduce* concerns about privacy:

- Privacy Policy, License Agreements, etc.: Describing what the system will/won't do with the data.
- Privacy Laws: Describing which national law the system is compliant with (e.g., HIPAA in the US, European privacy laws).
- Anonymizing all data: Ensuring that none of the data has any personal identifiers.
- Technical Details: Describing the algorithms/source code of the system in order to achieve higher trust (e.g., encryption of data).

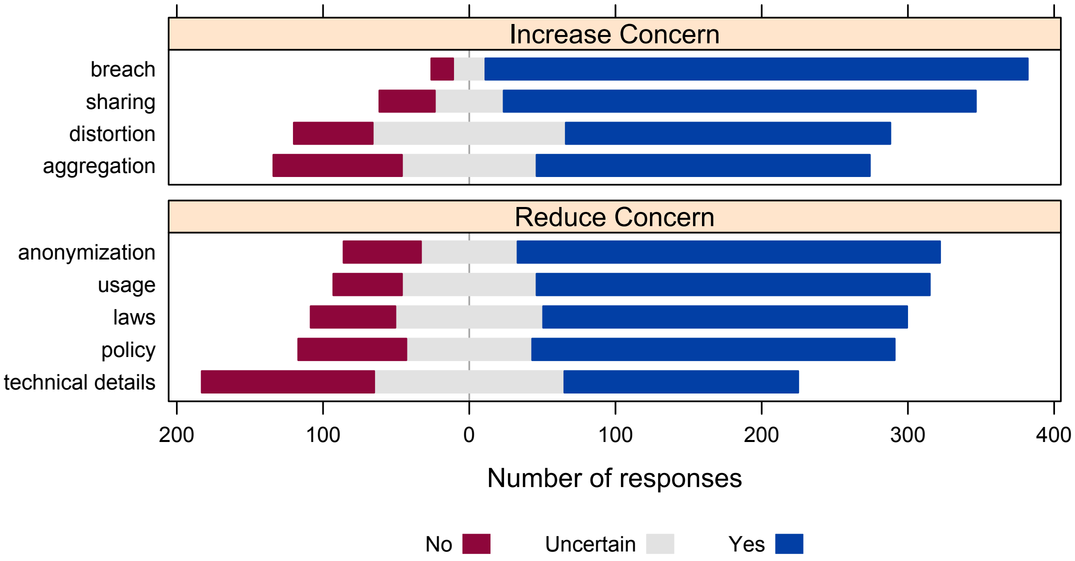


Figure 2.1: What increases and reduces privacy concerns?

- Details on usage: Describe, e.g., in a table how different data are used.

Figure 2.1 shows the overall answers for both questions. In the figure, each answer option is sorted by the number of “Yes” respondents. Most respondents agreed that the biggest privacy concerns are data breaches and data sharing. There is disagreement about whether data distortion and data aggregation would increase privacy concerns. To check if these results are statistically significant, we ran Z-tests for equality of proportions. This would help us validate, for example, if there is a statistically significant difference in the number of respondents who said “Yes” for two different options. The results for *increasing* concerns about privacy are shown in Table 2.2. For all of these tests, the null hypothesis is that a similar fraction of the respondents chose “Yes” for both options. The results show that the concerns about data breaches and data sharing are significantly higher than data aggregation and data distortion ($p \leq 1.231e^{-12}$).

Hypothesis 1: People are more concerned about the security aspects of privacy, in particular, about data sharing and data breaches than data distortion and data aggregation.

For *reducing* concerns, respondents consider technical details the least effective option (with p-values ranging from $7.218e^{-10}$ for comparing to policy to $2.2e^{-16}$ for comparing to anonymization).

Table 2.2: What *increases* privacy concerns? For each privacy concern, $X > Y$ indicates that X is a bigger concern than Y for the respondents. We used the Z-test for equality of proportions and only statistically significant results for $p < 0.01$ are shown.

Privacy concerns	p-values
Sharing > Aggregation	$p = 1.231e^{-12}$
Sharing > Distortion	$p = 6.036e^{-14}$
Breach > Aggregation	$p < 2.2e^{-16}$
Breach > Distortion	$p < 2.2e^{-16}$

Respondents think that anonymization is the most effective option for mitigating privacy concerns and significantly better than privacy laws ($p = 0.003$) and privacy policy ($p = 0.002$). There is, however, no statistically significant difference between anonymization and providing usage details ($p > 0.15$). The remaining three options (privacy policy, privacy laws, and usage details) had similar responses and none of their combinations for the Z-test yielded statistically significant results for $p < 0.01$.

Hypothesis 2: There is less agreement on the best measures for mitigating privacy concerns.

2.3.2 Qualitative Feedback

From the 408 respondents, we collected 135 comments on our open questions about additional privacy concerns and measures to reduce them. We analyzed these comments manually in three steps. First, we read each comment and annotated it with a few keywords. Thereby, we tried to reuse the keywords whenever possible. Second, we unified and grouped the keywords into topics, making sure that no important comments are lost. Finally, we read the comments again and assigned each of them to the identified topics.

2.3.2.1 Additional Privacy Concerns

We collected 66 comments on additional privacy concerns. 15 comments were useless as they just repeated the standard response options, were not understandable, or without content (e.g., “no comment”, or “nothing more”). The remaining 51 comments gave interesting insights, which can be grouped into the following topics:

Authorities and intelligent services: 13 respondents mentioned authorities and intelligent services as an additional privacy concern. One wrote: "Government access is not exactly a data breach, but still a privacy concern". Another commented: "anyway there is prism". It is important to mention that about half of the responses were collected after the NSA PRISM scandal [35, 48].

APIs, program correctness, and viruses: Nine respondents mentioned concerns related to the program behavior, including malicious programs and viruses. Respondents also mentioned that privacy concerns are "transmitted" through the application programming interfaces of the tools collecting data. One respondent wrote: "Sharing data over API" while others mentioned specific systems such as Google Analytics or Facebook API. Three respondents specifically pointed the correctness of privacy implementation as a specific concern.

Unusable and nontransparent policies: Seven users complained about unusable privacy implementations with unclear, nontransparent policies. These respondents were concerned because most users simply do not know which data is being collected about them and for what purposes. One respondent wrote: "Companies and software developers shield themselves [. . .] by making consumers agree on a long, convoluted, and often a hard to understand, hard to read [. . .] policy. Companies know that people do not read them a tactic on which they are banking". Another gave a more concrete example: "Sometimes sharing sensitive data is activated by default in applications (unaware users would leave it so)". One respondent wrote: "Transparency and letting the user choose make a huge difference. Maybe not in the beginning and maybe not for all users but definitely for a specific user group".

Intentional or unintentional misuse: At least seven respondents mentioned different forms of misusing the data as main concerns. This includes commercial misuse such as making products of interest more expensive, but it could also be misused for social and political purposes. Apart from abusing the data to put pressure on users, respondents mentioned using fake data to manipulate public opinions or inferencing sensitive information about groups of people and minorities. One respondent wrote: "Whenever something happen [sic] the media uses their data accessible online to 'sell' this person as good or evil".

Lack of control: Seven respondents mentioned the lack of control and in particular, options to delete data collected about them as their main concern. One wrote: "if we agree to give the data, we are not able anymore to revise this decision and delete the data. Even if the service confirms the deletion, we don't have any mean of control". Another respondent explicitly mentioned the case where companies owning their data are bankrupt or sold and in this case, the privacy of their data is also lost: "Company A has a decent privacy policy, Company B acquires the company, and in doing so, now has access to Company A's data".

Combined data sources: Five respondents explicitly mentioned combining data about users from different sources as a main privacy concern. In most cases, this cannot be anticipated when developing or using a single system or a service. One respondent wrote: "It's difficult to anticipate or assess the privacy risk in this case". Another claimed: "Continuous monitoring combined with aggregation over multiple channels or sources leads to complex user profiling. It's disturbing to know that your life is monitored on so many levels".

Collecting and storing data: Five respondents wrote that collecting and storing data is, on its own, a privacy concern. In particular, respondents complained about too much data being collected about them and stored for too long time. One respondent mentioned: "The sheer amount of cookies that are placed on my computer just by landing on their website". Another claimed: "Collecting the data and storing for a long period of time is seen more critical than just collecting".

Other issues: Three respondents mentioned problems with the legal framework and in particular, the compatibility of laws in the developer and user countries. Three respondents said that in some cases there is no real option to not use a system or service, e.g., due to a "social pressure as all use Facebook" or since "we depend on technology".

2.3.2.2 Suggestions for Reducing Privacy Concerns

In total, 69 respondents answered the open question on additional measures to reduce user concerns about privacy. Ten of these answers either repeated the standard options or were useless. The remaining 59 comments showed more convergence in the opinion than the comments on the additional concerns, possibly because this question was more concrete. The suggestions can be grouped into the following measures:

Easy and fine-grained control over the data, including access and deletion: 17 respondents recommended allowing the users to easily access and control the collected and processed data about them. In particular, respondents mentioned the options of deactivating the collection and deleting the data. One respondent wrote: "To alleviate privacy concerns, it should be possible to opt out of or disagree with certain terms". Another wrote: "Allow users to access a summary of all the data stored on their behalf, and allow them to delete all or part of it if they desire". The respondents also highlighted that this should be simple and easy to do and embedded into the user interface at the data level.

Certification from independent trusted organizations: 14 respondents suggested introducing a privacy certification mechanism by independent trusted authorities. A few also suggested the continuous conduction of privacy audits similar to other fields such as safety and banking. Respondents also suggested that the results of the checks and audits should be made public to increase the pressure on software vendors. One respondent even suggested "having a privacy police to check on how data is handled".

Transparency and risk communication, open source: 13 respondents mentioned increased transparency about the collection, aggregation, and sharing of the data. In particular, respondents mentioned that the risks of misusing the data should be also communicated clearly and continuously. Three respondents suggested that making the code open source would be the best approach for transparency. One wrote: "tell users (maybe in the side-bar) how they are being tracked. This would educate the general public and ideally enable them to take control of their own data". The spectrum of transparency was from the data being collected to physical safety measures of servers and qualifications of people handling data to how long the data is stored.

Period and amount of data: 11 respondents recommended always limiting and minimizing the amount of data and the period of storage, referring to the principle of minimality. The period of time for storing the data seems to be crucial for users. One wrote: “Not allowing users data being stored in servers. Just maintaining them in the source”.

Security and encryption: We noted that respondents strongly relate privacy issues to information security. At least seven suggested security measures, mainly complete encryption of data and communication channels.

Trust and education: Seven respondents mentioned building trust in the system and vendor as well as education of users on privacy as effective means to reduce privacy concerns.

Short, usable, precise and understandable description, in the UI: At least six respondents mentioned increasing the usability to access data and policy as an important measure to reduce privacy concerns. One wrote: “the disclaimer should be directly accessible from the user interface when conducting a function which needs my data”. Another respondent wrote: “short understandable description and no long complicated legal text”.

2.3.3 Criticality of Different Types of Data

To get a deeper insight into the privacy criticality of different types of data, we asked respondents to rate the following types of data on a 5-point semantic scale ranging from “Very critical” to “Uncritical”.

- Content of documents (such as email body)
- Metadata (such as date)
- Interaction (such as a mouse click to open or send an email)
- User location (such as the city from where the email was sent)
- Name or personal data (such as email address)

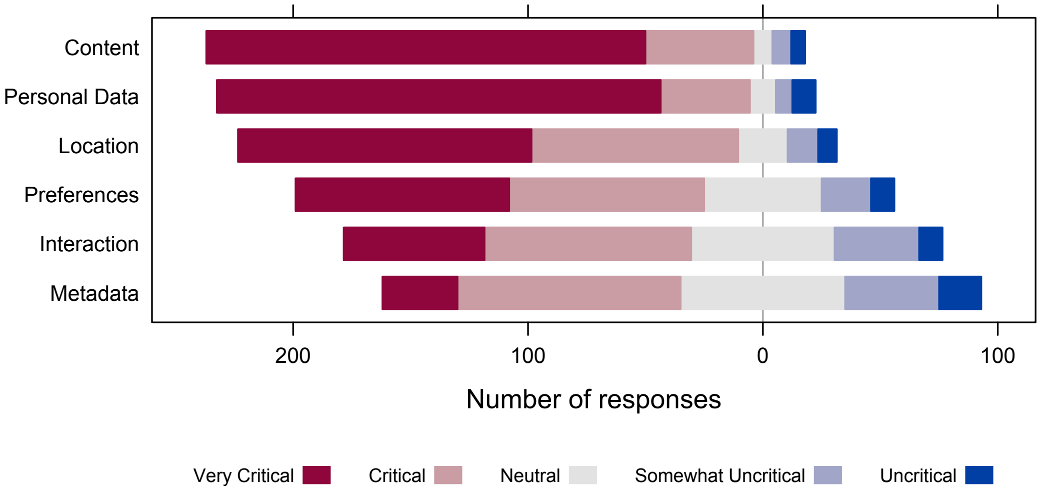


Figure 2.2: How *critical* would you rate the collection of the following data?

- User preferences (such as inbox or email settings)

The results are shown in Figure 2.2. Respondents chose content as most critical, followed by personal data, location, preferences, and interaction and metadata are the least critical as far as privacy is concerned.

We used Welch's Two Sample t-test to compare if the difference among the different types of data is statistically significant. The null hypothesis was that the difference in means was equal to zero. Table 2.3 summarizes the results. It shows, for example, that there is no statistically significant difference between content and personal data. On the other hand, there is a statistically significant difference between content and location for $p < 0.01$.

Hypothesis 3: People are more concerned about content and personal data than interaction and metadata.

2.3.4 Giving up Privacy

We asked respondents if they would accept *less* privacy for the following:

- Monetary discounts (e.g., 10% discount on the next purchase)
- "Intelligent" or added functionality of the system (such as the Amazon recommendations)

Table 2.3: The significance in the difference between the *criticality* of collecting different data. p-values: '+ + +' for $p < e^{-11}$, '++' for $p < e^{-6}$, '+' for $p < 0.01$, and ' ' for $p > 0.01$. The rows and columns are ordered from most to least critical. For each cell, t-tests compare if the difference in criticality is statistically significant. For example, the difference between interaction and content is statistically significant for $p < e^{-11}$.

	Content	Personal Data	Location	Preferences	Interaction	Metadata
Content	–					
Personal Data		–				
Location	+	+	–			
Preferences	+++	++	+	–		
Interaction	+++	+++	++	+	–	
Metadata	+++	+++	+++	++	+	–

- Fewer advertisements

For each option, the respondents could answer using a 3-point semantic scale having options: "Yes", "Uncertain", and "No". The results are shown in Figure 2.3.

36.7% of the respondents said they would accept less privacy for added functionality of the system while only 20.7% and 13.7% would accept less privacy for monetary discounts and fewer advertisements respectively. Added functionality seems to be the most important reason to accept less privacy. These results are statistically significant using the Z-test for equality of proportions ($p < 3.882e^{-5}$ for monetary discounts and $p < 1.854e^{-9}$ for fewer advertisements). It is important to note that *less than half* of the respondents would accept less privacy for added functionality of the system.

Previous studies, such as the one conducted by Acquisti et al. [3], have shown, however, that people's economic valuations of privacy vary significantly and that people *do* accept less privacy for monetary discounts. This contrast in results might be due to a difference between people's opinion and their actual behavior.

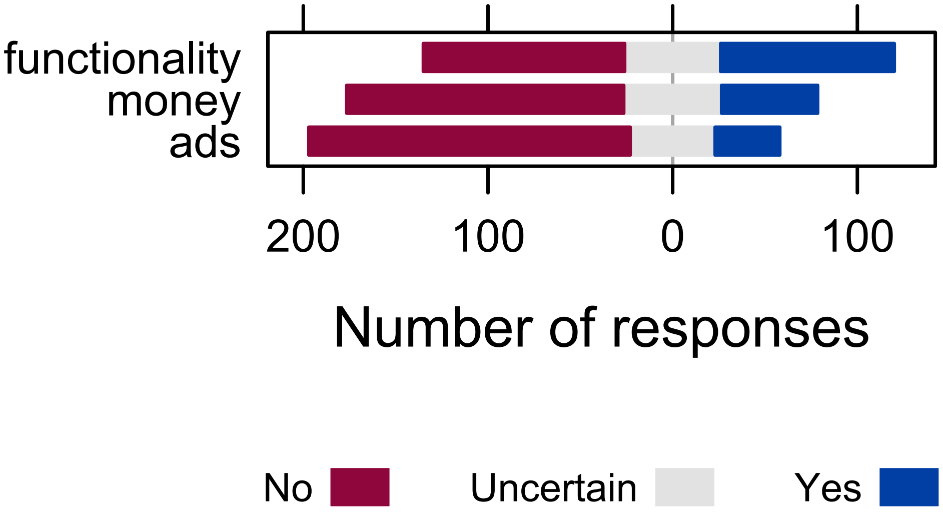


Figure 2.3: Would users accept *less* privacy for the following?

Hypothesis 4: There are different groups of opinions about accepting less privacy for certain benefits. The largest group of users say that they are not inclined to give up privacy for additional benefits. However, their actual behavior might be different.

2.4 Developer vs User Perceptions

The results from the previous section describe the broad concerns for all respondents of our survey. In this section, we report on the important results from a differential analysis between two groups of respondents: developers (267 out of 408) versus users of software systems (141 out of 408). We used Z-tests for equality of proportions for the rest of this section, unless otherwise noted.

2.4.1 Privacy Concerns

Data distortion: 49.1% of developers believe that data distortion is an important privacy concern. The percentage of users, on the other hand, is 64.5%. The difference between these two groups is statistically significant ($p = 0.003$).

Data aggregation: 52.1% of developers believe that data aggregation is an important privacy concern. The percentage of users, on the other hand, is 63.1%. The difference between them is statistically significant ($p = 0.04185$). It seems that developers trust their systems more than users when it comes to wrong interpretation of sensitive data.

Data criticality: Developers believe that "name and personal data" ($p = 0.038$) and "interaction" ($p = 0.082$) are more critical for privacy compared to users. On the other hand, for the remaining four categories (content, location, preferences, metadata), there is no statistically significant difference between the perceptions of developers and users ($p > 0.2$ for all). We used Welch's Two Sample t-test here.

Less privacy for added functionality: A larger fraction of developers (43.3%) would accept less privacy for added or intelligent functionality of the system compared to 31.2% of users ($p = 0.002$).

Hypothesis 5: Developers are more concerned about interaction, name, and personal data whereas users are more concerned about data distortion and data aggregation.

2.4.2 Measures to Reduce Concerns

Developers and reducing concerns: A larger fraction of developers (71.2%) feel that data anonymization is a better option to reduce privacy concerns as compared to privacy policies or privacy laws (both, 56.9%) ($p = 0.0006$). 66.3% of developers prefer providing details on data usage for mitigating privacy concerns compared to privacy policies (56.9%) ($p = 0.03$).

Similarly, 20.2% of developers feel that privacy policies will *not* reduce privacy concerns whereas only 11.2% feel that providing details on data usage will *not* be beneficial ($p = 0.004$).

Users and reducing concerns: In contrast, for users, there is no statistically significant difference between their perception on privacy policies, laws, anonymization, and providing usage details. ($0.6 < p$ for all combinations).

Hypothesis 6: Developers prefer anonymization and providing usage details as measures to reduce privacy concerns. Users, on the other hand, do not have a strong preference.

2.5 The Role of Geography

In this section, we present the results of the differential analysis based on the geography of respondents. We asked respondents to specify with which region of the world they identify themselves. The options were North America, South America, Europe, Asia/Pacific, Africa, and other. Since we have only seven responses from South America and Africa combined, we focus on the differences between the others. We used the Z-test for equality of proportions for the rest of this section, unless otherwise noted.

Data criticality: We asked respondents to rate the criticality of the different types of data for privacy (i.e., content of documents, metadata, interaction, user location, name or personal data, user preferences) using a semantic scale from 1–5, with 1 being “Very Critical” and 5 being “Uncritical”.

There is a statistically significant difference between respondents from North America, Europe, and Asia/Pacific. We used Welch’s Two Sample t-test to compare the ratings given by respondents. Respondents in North America think that all types of data are *less critical* (overall mean across the six types of data is 2.31) than respondents in Europe (overall mean is 1.87) for $p = 3.144e^{-8}$.

Similarly, respondents from North America think all items are less critical that those in Asia/-Pacific (overall mean: 2.01) for $p = 0.037$. On the other hand, there is no statistically significant difference between respondents in Europe and Asia/Pacific ($p > 0.28$).

Less privacy for added functionality: A larger fraction of respondents in Europe (50.6%) claim that they would *not* give up privacy for added functionality. In North America, on the other hand, this fraction is 24.1%. The difference between the two regions is statistically significant ($p = 0.0001$).

Hypothesis 7: People from North America are more willing to give up privacy and feel that different types of data are less critical for privacy compared to people from Europe.

Concerns about data sharing versus data distortion: A larger fraction of respondents in North America (88.9%) feel that data sharing is a concern compared to 46.3% for data distortion ($p = 6.093e^{-6}$). On the other hand, there is no statistically significant difference among respondents in Asia/Pacific ($p > 0.67$).

Concerns about data sharing versus data breach: In Europe, a larger fraction of the respondents (94.3%) are concerned about data breaches as compared to 76.4% for data sharing. The difference is statistically significant ($p = 5.435e^{-6}$). On the other hand, there is no statistically significant difference among respondents in North America ($p > 0.12$).

Laws versus usage details: In Europe, a larger fraction of respondents (75.9%) feel that providing details on how the data is being used will reduce privacy concerns as opposed to 58.1% who feel that privacy laws will be effective ($p = 0.00063$). On the other hand, there is no statistically significant difference among respondents in North America, where the percentage of respondents are 67.9% and 64.2% respectively ($p > 0.43$).

Usage details versus privacy policies: A larger fraction of respondents in Europe (75.9%) feel that providing usage details can mitigate privacy concerns compared to 63.2% for using a privacy policy ($p = 0.015$). On the other hand, there is no statistically significant difference among respondents in North America ($p > 0.32$).

Hypothesis 8: People from Europe feel that providing usage details can be more effective for mitigating privacy concerns than privacy laws and privacy policies whereas people from North America feel that these three options are equally effective.

2.6 Discussion

We discuss our results, potential reasons, and the implications for software developers and analysts. We also reflect on the limitations and threats to validity of our results.

2.6.1 Privacy Interpretation Gaps

Data privacy is often an implicit requirement: everyone talks about it but no one specifies what it means and how it should be implemented. This topic also attracts the interests of different stakeholders including users, lawyers, sales people, and security experts, which makes it even harder to define and implement. One important result from our study is that while almost all respondents agree about the importance of privacy, the understanding of the privacy issues and the measures to reduce privacy concerns are divergent. This calls for an even more careful and distinguished analysis of privacy when designing and building a system.

Our results from Sections 2.4 and 2.5 show there is a definite gap in privacy expectations and needs between users and developers and between people from different regions of the world. Developers have assumptions about privacy, which do not always correspond to what users need. Developers seem to be less concerned about data distortion and aggregation compared to users. It seems that developers trust their systems more than users when it comes to wrong interpretation of privacy critical data. Unlike users, developers prefer anonymization and providing usage details for mitigating privacy concerns. If the expectations and needs of users do not match those of developers, developers might have wrong assumptions and might end up making wrong decisions when designing and building privacy-aware software systems.

In addition, privacy is not a universal requirement as it appears to have an internationalization aspect to it. Different regions seem to have different concrete requirements and understanding for privacy. Our results confirm that there exist many cultural differences between various regions of the world as far as privacy is concerned. The recent NSA PRISM scandal has also brought these differences into sharp focus. A majority of Americans considered NSA's accessing personal data to prevent terrorist threats more important that privacy concerns [35]. In contrast, there was widespread "outrage" in Europe over these incidents [48]. It also led to an article in the New York Times by Malte Spitz, a member of the German Green Party's executive committee, titled "Germans Loved Obama. Now We Don't Trust Him" [140]. These differences, both in terms of laws and people's perceptions, should be considered carefully when designing and deploying software systems.

We think that privacy should become an explicit requirement, with measurable and testable criteria. We also think that privacy should also become a main design criteria for developers as software systems are collecting more and more data about their users [37]. To this end, we feel that there is a need to develop a standard survey for privacy that software teams can customize and reuse for their projects and users. Our survey can be reused to conduct additional user studies on privacy for specific systems. Our results can also serve as a benchmark for comparing the data. This can help build a body of knowledge and provide guidelines such as best practices.

2.6.2 The Security Dimension of Privacy

We think that people are more concerned about data breaches and data sharing as there have been many recent instances that have received a lot of media coverage. To list a few recent examples, Sony suffered a massive data breach in its Playstation network that led to the theft of personal information belonging to 77 million users [13]. One hundred and sixty million credit card numbers were stolen and sold from various companies including Citibank, the Nasdaq stock exchange, and Carrefour [122]. The Federal Trade Commission publicly lent support to the "Do-Not-Track" system for advertising [9]. Compared to these high-profile stories, we feel that there have been relatively few "famous" instances of privacy problems caused by data aggregation or data distortion yet.

There is a large body of research that has advanced the state-of-the-art in security (encryption) and authorization. One short-term implication for engineers and managers is to systematically implement security solutions when designing and deploying systems that collect user data, even if it is not a commercially or politically sensitive system. This would significantly and immediately reduce privacy concerns. For the medium-term, more research should be conducted for deployable data aggregation and data distortion solutions.

As far as mitigating privacy concerns, our results show that there is more disagreement. We believe that the reason for this is that online privacy concerns are a relatively recent phenomenon. Due to this, people are not sure which approach works best and might be beneficial in the long run.

2.6.3 Privacy Framework

We feel that an important direction for software and requirements engineering researchers is to develop a universal, empirically grounded framework for collecting, analyzing, and implementing privacy requirements. This study is the first building block towards such a framework. Some of the lessons learned from our study can be translated into concrete qualities and features, which should be part of such a framework. This includes:

- **Anonymization**: This is perhaps the most well-known privacy mitigating technique and seems to be perceived as an important and effective measure by both users and developers. Developers should therefore use anonymization algorithms and libraries.

- **Data usage**: Although anonymization is perceived as the most effective measure for addressing privacy concerns, this is currently not practical as approaches like differential privacy are computationally infeasible [103, 146]. In such situations, it is better to provide users with data usage details and make these more transparent and easier to understand. Our findings show that there is no statistical difference between anonymization and providing usage details as far as users are concerned. Thus, in terms of future research, it is perhaps better to focus on improving techniques for providing data usage details rather than (or in addition to) making anonymization computationally feasible.

- **Default encryption**: As users are mainly concerned about the loss and abuse of their data, systems collecting user data should implement and activate encryption mechanism for storing and transmitting these data. In Facebook, e.g., the default standard communication protocol should be HTTPS and not HTTP.

- **Fine-grained control over the data**: Users become less concerned about privacy if the system provides a mechanism to control their data. This includes activating and deactivating the collection at any time, the possibility to access and delete the raw and processed data, and define who should have access to what data.

- **Interaction data first**: Users have a rather clear preference of the criticality of the different types of data collected about them. Therefore, software researchers and designers should first try to implement their systems based on collecting and mining interaction data instead of content of files and documents. Research has advanced a lot in this field in, especially, recommender systems [92].

- **Time and space-limited storage**: The storage of data about users should be limited in time and space. The location where the data is stored is an important factor for many respondents. Therefore, systems should provide options for choosing the location of storing privacy sensitive data.

- **Privacy policies, laws, and usage details**: Users rated all these options as equally effective for mitigating their privacy concerns. Therefore, developers could utilize any of these options, thus giving them better flexibility in the design and deployment of software systems.

2.6.4 Limitations and Threats to Validity

There are several limitations to our study, which we discuss in this section. The first limitation is a potential selection bias. Respondents who volunteered to fill out our survey were self-selected. Such selection bias implies that our results are only applicable to the volunteering population and may not necessarily generalize to other populations. The summaries have helped us identify certain trends and hypotheses and these should be validated and tested by representative samples, e.g., for certain countries. In contrast, the differential analysis (also called pseudo-experimentation) conducted within our set of respondents, enabled us to identify statistically significant relationships and correlations. Hence, many of our results deliberately focus on correlations and cross-tabulations between different populations.

As for internal validity, we are aware that by filling out a brief survey, we can only understand a limited amount of concerns that the respondents have in mind. Similarly, the format and questions of the survey might constrain the expressiveness of some of the respondents. We might have missed certain privacy concerns and measures to reduce concerns by the design of the survey. We tried to mitigate this risk by providing open-ended questions that respondents could use to express additional aspects they had in mind. Moreover, we designed the survey in a highly iterative process and tested it in dry runs to ensure that all options are understandable and that we did not miss any obvious option.

As with any online survey, there is a possibility that respondents did not fully understand the question or chose the response options arbitrarily. We conducted several pilot tests, gave the option to input comments, and the incompletion rate is relatively small. We included a few validation questions and we only report responses in this project from respondents who answered these questions correctly. We also provided two versions of the survey, in English and German, to make it easier for non-native speakers.

In spite of these limitations, we managed to get a large and diverse population that filled out our survey. This gives us confidence about the overall trends reported in this project.

2.7 Related Work

There has been a lot of research about privacy and security in different research communities. We summarize the important related work focussing on usability and economic aspects of privacy, anonymization techniques, and work from the software and requirements engineering community.

Many recent studies on online social networks show that there is a (typically, large) discrepancy between users' intentions for what their privacy settings should be versus what they actually are. For example, Madejski et al. [91, 94] report in their study of Facebook that 94% of their participants ($n = 65$) were sharing something they intended to hide and 85% were hiding something that they intended to share. Liu et al. [91] found that Facebook's users' privacy settings match their expectations only 37% of the time. A recent longitudinal study by Stutzman et al. [143] shows how privacy settings for Facebook users have evolved over a period of time. These studies have focused on privacy settings in a specific online system whereas our study was designed to be agnostic to any modern system collecting user sensitive data. Further, the main contribution of these studies is to show that there is a discrepancy between what the settings are and what they should be and how settings evolve over time. Our study aims to gain a deeper understanding of what the requirements are and how they change across geography and depending on software development experience.

Fang and LeFevre [55] proposed an automated technique for configuring a user's privacy settings in online social networking sites. Paul et al. [118] present using a color coding scheme for making privacy settings more usable. Squicciarini, Shehab, and Paci [142] propose a game-theoretic approach for collaborative sharing and control of images in a social network. Toubiana et al. [148] present a system that automatically applies users' privacy settings for photo tagging. All these papers propose new approaches to make privacy settings "better" from a user's perspective (i.e., more usable and more visible). Our results help development teams decide when and which of these techniques should be implemented. We focus more on a broader requirements and engineering perspective of privacy than on a specific technical perspective.

There has been a lot of recent work on the economic ramifications of privacy. For example, Acquisti et al. [3] (and the references therein) conducted a number of field and online experiments to investigate the economic valuations of privacy. In Section 2.3.4, we discussed whether users would give up privacy for additional benefits like discounts or fewer advertisements. Our study complements and contrasts the work of Acquisti et al. as described earlier.

There has also been a lot of work about data anonymization and building accurate data models for statistical use (e.g., [5, 49, 88, 121, 159]). These techniques aim to preserve certain properties of the data (e.g., statistical properties like average) so they can be useful in data mining while trying to preserve privacy of individual records. Similarly, there has also been work on anonymizing social networks [21] and anonymizing user profiles for personalized web search [175]. The broad approaches include aggregating data to a higher level of granularity or adding noise and random perturbations. There has been research on breaking the anonymity of data as well. Narayanan and Shmatikov [107] show how it is possible to correlate public IMDb data with private anonymized Netflix movie rating data resulting in the potential identification of the anonymized individuals. Backstrom et al. [12] and Wondracek et al. [166] describe a series of attacks for de-anonymizing social networks.

Also in the software engineering community, recent papers on privacy mainly focused on data anonymization techniques. Clause and Orso [33] propose techniques for the automated anonymization of field data for software testing. They extend the work done by Castro et al. [32] using novel concepts of path condition relaxation and breakable input conditions resulting in improving the effectiveness of input anonymization. Taneja et al. [145] and Grechanik et al. [65] propose using k-anonymity [144] for privacy by selectively anonymizing certain attributes of a database for software testing. They propose novel approaches using static analysis for selecting which attributes to anonymize so that test coverage remains high. Our work complements these papers as respondents in our study considered anonymization an effective technique for mitigating privacy concerns and these techniques could be used as part of a privacy framework.

There have been some recent papers on extracting privacy requirements from privacy regulations and laws [27, 28]. These could be part of the privacy framework as well and help in reducing the impact due to cultural differences for privacy. While this work focus on legal requirements, we focus on the users' understanding of privacy and how it differs from developers' views. A few recent papers have also discussed privacy requirements, mainly in the context of mobile applications. Mancini et al. [95] conducted a field study to evaluate the impact of privacy and location tracking on social relationships. Tun et al. [150] introduce a novel approach called "privacy arguments" and use it to represent and analyze privacy requirements in mobile applications. Omoronyia et al. [112] propose an adaptive framework using privacy aware requirements, which will satisfy runtime privacy properties. Our focus is broader than the first two papers as we don't limit our scope to mobile applications;

nonetheless, many of our findings would apply directly. Our work is complementary to the last paper where our findings could be used as part of the adaptive framework.

Finally, many authors in the software engineering and requirements engineering communities mention privacy in the discussion or challenges section of their papers (e.g., [30, 92, 93, 104]. But in most cases, there is little evidence and grounded theory about what, how, and in which context privacy concerns exist and what the best measures for addressing them are. Our study helps in clarifying these concerns and measures as well as comparing the different perceptions of people.

2.8 Conclusion

In this project, we conducted a study to explore the privacy requirements for users and developers in modern software systems, such as Amazon and Facebook, that collect and store data about the user. Our study consisted of 408 valid responses representing a broad spectrum of respondents: people with and without software development experience and people from North America, Europe, and Asia. While the broad majority of respondents (more than 91%) agreed about the importance of privacy as a main issue for modern software systems, there was disagreement concerning the concrete importance of different privacy concerns and the measures to address them. The biggest concerns about privacy were data breaches and data sharing. Users were more concerned about data aggregation and data distortion than developers. As far as mitigating privacy concerns, there was little consensus on the best measure among users. In terms of data criticality, respondents rated content of documents and personal data as most critical versus metadata and interaction data as least critical.

We also identified difference in privacy perceptions based on the geographic location of the respondent. Respondents from North America, for example, consider all types of data as less critical for privacy than respondents from Europe or Asia/Pacific. Respondents from Europe are more concerned about data breaches than data sharing whereas respondents from North America are equally concerned about the two.

We also gave some insight into a framework and a set of guidelines on privacy requirements for developers when designing and building software systems. This is an important direction for future research and our results can help establish such a framework, which can be a catalog of privacy concerns and measures, a questionnaire to assess and fine-tune them, and perhaps a library of reusable privacy components.

Finally, as far as societal computing is concerned, this project can be a good starting point for exploring the various tradeoffs. The study design and artifacts can serve as a template for conducting further studies that try to gain insights into what the user concerns are for the different tradeoffs in societal computing. Typically, it is hard to know how different users with varying backgrounds, diversity in geographic locations, and experience in software development will respond to different societal concerns like cultural differences, laws, green computing, etc. Using a survey similar to the one described in this chapter can help us achieve deeper insight into what the main concerns are and how best to mitigate these concerns.

Further, generalizing this survey to other domains can help us understand and quantify these tradeoffs in a more rigorous manner. The big take-away, as far as this project is concerned, is that privacy is treated very differently by different populations. This might likely be the case for the other domains of societal computing. If that is indeed the case, gaining a deeper insight would be the first step towards trying to address these problems and the tradeoffs therein.

Chapter 3

Making Privacy Understandable — Crowdsourcing Privacy Settings

The results of the previous chapter show us that for mitigating privacy concerns, end-users are equally satisfied with more transparency and details on how their data is being used. Complex techniques like anonymization are not necessary. In this chapter, we build upon the results from our survey — we present a tool that provides users with more transparency for controlling their privacy settings on online social networks like Facebook.

Imagine systems like Facebook which have complex user interfaces and that change the privacy settings often. Numerous studies have shown that the privacy settings of users are not what they intended them to be [77, 91, 94]. Our approach towards solving this problem is to use Crowdsourcing. For example, if most of my friends have their home address as private, maybe that would indicate to me that I should change my settings to make my address private as well. Thus, for any user, we can analyze what their settings are using the API and compare this to canonical settings. This canonical list can be generated in multiple ways as preferred by the users — it could be a list of "trusted" friends, it could be all friends of that user, it could be all of my friends, etc. This comparison will be shown to the user via various visualizations (entire list of differences, a "score" between 0 and 1 that tells the users how private/public they are compared to the canonical list, etc.). This would be similar to the "Gullibility Factor" mentioned earlier. Users can now choose to keep their settings the same as they were before, modify them manually, or just mimic the settings from the canonical list.

Even if the privacy settings are what they intended, maybe the users can benefit from seeing what others do on the social networking website. Note, an important issue here is trust: I, typically, would not trust any random friend or user of the website; my real life trust of a set of friends will impact who I want to mimic.

3.1 Introduction

As social network platforms, such as Facebook, have increased the amount of personal data they contain, the ease with which individual users can understand and control their own data is a growing concern [169]. Current privacy settings are often complicated and difficult to navigate [62]. For example, Facebook obfuscates privacy settings by splitting privacy controls across several seemingly unrelated portions of the application. The problem is made worse by occasional unannounced changes that are instituted without giving users the option to opt-out before the change takes effect [56].

Previous research efforts have focused on how well users understand their privacy. Some of this research has shown a disparity between what users intend when configuring their privacy settings and the reality of what their settings represent [90, 91]. Other research has introduced techniques aimed at making privacy more understandable and easier to control. There have been several approaches, including introducing separate third-party software to store and manage a social network user's content independently of the social network service [147], providing access controls that enable social network users to segregate risky connections [18], and creating automated tools for classifying every connection into labeled groups and then applying label-based privileges to these groups [55]. However, none of these techniques allowed users to see and understand their privacy settings alongside the privacy settings from other individuals or peer groups. Because social networks are designed for sharing, a contextualized view of privacy (seeing one's privacy settings alongside the settings of others) is easier to understand than an isolated view. This is our approach. We use crowd sourced data to give users more information with which to make decisions on how they would like to set their privacy.

This project aims to introduce a crowd sourced technique for improving social network users' understanding of privacy. To this end we conducted a study which included an online survey for which we received 59 valid responses. As part of the survey, we presented users with three alternative privacy control options: 1. a 3-option model (easy/medium/hard), 2. a survey based model, and 3. a crowd sourced model. After the survey, we performed follow-up interviews with 10 respondents in order to confirm survey responses with additional qualitative feedback, and to determine if user understanding of privacy could be improved with data visualizations. The visualizations used were produced with data collected from 512 Facebook users over a 17 month period.

The results of the survey show that users tended to prefer the 3-option and survey-based models. During the follow-up interviews users were presented with mock-ups of the three models and asked to rate them. The results show that users preferred all of the alternative models over the current controls, and in particular, the survey based and crowd sourced models were the most popular options. We believe the change-of-preference reflects the lack of user familiarity with crowd sourced models. Additionally, we found that 70% of users found data visualizations useful when configuring privacy settings.

3.2 Approach

The motivation for this study was to improve user understanding of privacy by addressing issues that currently exist with social network's privacy controls: persistence (how long a user-defined setting remains constant), validation (how easily a user is able to confirm their settings), and comparison (the ability for users to see their privacy settings alongside the settings of other users and groups). In this section, we outline our approach.

3.2.1 Crowdsourcing Privacy

In order to improve user understanding of privacy through a crowd sourced approach, we created a set of visualizations that summarize a user's privacy settings and provide a comparative view with the settings from other users. These visualizations can be split into two different categories: Individual Visualizations, containing only data for a single user, and Contextualized Visualizations, showing a single users data in the context of the data of multiple users. The following subsections describe the dataset that was used and the visualizations that were created.

3.2.1.1 Dataset

Our dataset contains the information of 512 Facebook users, collected since May 1st 2012. The data was collected using an automated tool that retrieved users' privacy information from the Facebook Graph API. On average, there were a total of 154 scans made per user. The number of scans varied slightly due to API outages and incomplete or badly formatted API responses. Each scan has data representing the amount of information in each category that a user is sharing, such as number of mutual friends, number of check-ins, number of photos, etc., for a total of 41 different categories. By collecting data across all categories for a single user, we have a measure of how much information that particular user is sharing. Similarly, by combining the data collected across users, we have a measure of how much each category is being shared.

3.2.1.2 Individual Visualizations

The first visualization (Figure 3.1) in this group, is a line graph displaying the data from each category for an individual user across time. This view is valuable to a user because it allows them to confirm their settings have not changed unexpectedly (persistence), and get an overview of their settings (validation). The y-axis of the visualization represents the magnitude of the returned data. For example, the number of friends a user has. The x-axis represents time, with each increment marking a single scan of the user's data. Each line in the visualization represents a different category. The categories are set apart with different colors and distinct shapes for the anchors. When data was unavailable for a category the value was mapped to 0. When the Facebook Graph API returned an error, the value was mapped to -10. This visualization is helpful for identifying changes over time for the different categories, and provides users with a history and current snapshot of what information is being shared.

The second visualization (Figure 3.2) in this group is a word cloud. The word cloud provides users with a quick and easily understood view of what categories they are sharing (validation and persistence). The size of the font for each category represents the the magnitude of the shared content (how many friends a user has). Categories for which a user shares no data have the smallest font. If an API error was returned the text for that category is striked out. An example of this is shown with the "inbox" in Figure 3.2. While the word cloud visualization provides users with an easily understood display of their individual privacy settings, it could also be used to display the data of a group of users. In this case, the size of the font would represent the number of users sharing a category, and the strike out would indicate that no users share the category. This would allow users to quickly understand how common it is for an category to be shared (comparison).

3.2.1.3 Contextualized Visualizations

The first visualization (Figure 3.3) in this group is a series of donut charts. This visualization shows an individual users content in the context of a larger group of users, thereby allowing them to easily see how their settings compare to the settings of the members of the larger group. Each donut represents all of the data collected from the 512 Facebook users for single category. The blue portion of the donut represents the percentage of users that share data for the category, the orange represents the percentage that do not have data for the category, and the gray represents the percentage for which the Facebook Graph API returned an error. API errors typically indicated the category had been restricted as policy or for which the user had restricted sharing. For the individual using this visualization, we indicated whether or not they shared category by fading out the category if it was not shared, and additionally adding a color-coded dot at the top left of the donut. This visualization is useful when determining what are the trending privacy settings for a specific group (in this case, the entire set of data that was collected from the 512 over the 17 period).

The second visualization (Figure 3.4) in this group displays data about how the settings for a single category have changed over time, for multiple users, and highlights the individual user within this data. The y-axis represents the magnitude of the data being shared (e.g., the number of friends), and the x-axis represents the scan number (e.g., 50 represents the 50th scan). Each line represents a single users data. The highlighted line represents the individuals users data among the larger group. Again, for cases where there was no data returned the value was mapped to 0, and for cases where an API error occurred the value was mapped to -10. This visualization provides the individual with a good measure of how much information they are sharing compared to the larger group (comparison). Additionally, this visualization is useful for detecting changes in privacy, or, in a broader context, “global” changes (persistence and validation). As can be seen in Figure 3.4, if the visualization shows abrupt changes to many users data simultaneously, it may indicate a policy change, widespread error, or other noteworthy event.

3.2.2 Alternative Tools

In order to determine if there are tools that are preferred over the currently available privacy configuration tools, we selected a list of three alternatives. These tools attempt to simplify user configuration of privacy by minimizing the amount of user input required to achieve the desired settings.

The first mechanism is a 3-option (easy/medium/hard) system, where users can preset a system wide default level of privacy from a list with only three options. The user could, for example, set the default level of privacy to “completely private,” “friends only,” or “completely public”. This mechanism enables users to quickly define a privacy level. The second system consists of a short survey where users are indirectly asked about their personal preferences, and the tool would later determine which are the best privacy settings for that user based on the responses. The third mechanism is a crowd sourced tool, where users can set their privacy level based off a particular individual or a group of their choosing.

For the three tools, in order to fully satisfy a user’s preferences, the settings determined would serve as a baseline and users were given the option to later tweak the settings using the currently available mechanisms.

3.3 Study Design

In this section we will describe the research questions and methods used to validate our approach.

3.3.1 Research Questions

The goal of this study is to examine methods for improving user understanding of privacy settings. In particular, we want to determine if crowd sourced data can be used to simplify comprehension of privacy settings. The Oxford English Dictionary defines privacy as the “Absence or avoidance of publicity or display; secrecy, concealment, discretion; protection from public knowledge or availability [1].” We use the term *privacy* specifically as it relates to data privacy for information stored on social networks. We use the term *privacy settings* as a reference to the end-user configurable settings managing the accessibility of data stored on social networks. By *understand*, we mean the knowledge a user has about what information they are and are not sharing on a social network.

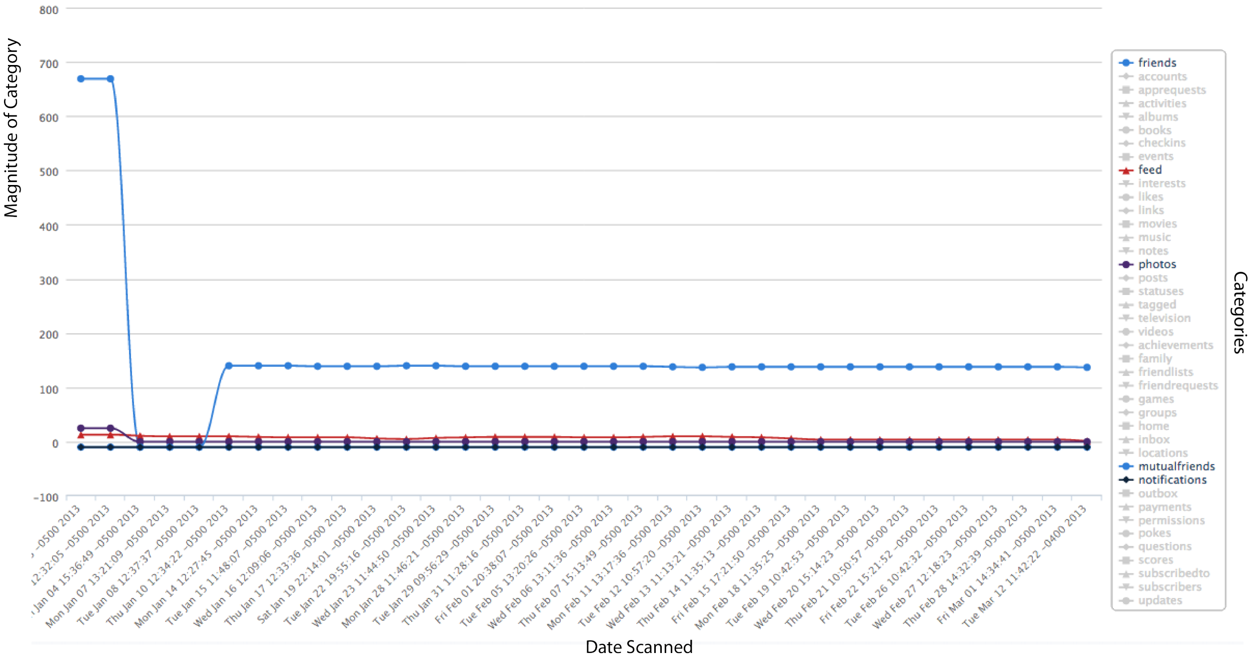


Figure 3.1: Individual visualization: Visualization of a single user's data from all categories presented over time.

We are not interested in whether or not users are aware of potential consequences that sharing information may entail.

We focused on the following research questions:

- **RQ 1:** Do users understand their current privacy settings? (Section 4)
- **RQ 2:** Can user understanding of privacy be improved using crowd sourced data? (Section 5)
- **RQ 3:** Are there tools for configuring privacy that are preferred over the currently provided tools? (Section 6)

3.3.2 Research Methods

Our study is designed to both qualitatively and quantitatively explore user understanding of privacy settings and preferences for privacy tools. As a quantitative approach, we created an online survey that consisted of 26 questions. Out of these, 16 were closed questions and respondents had to choose and answer from a given list of options. These questions consisted of 3-point and 5-point semantic

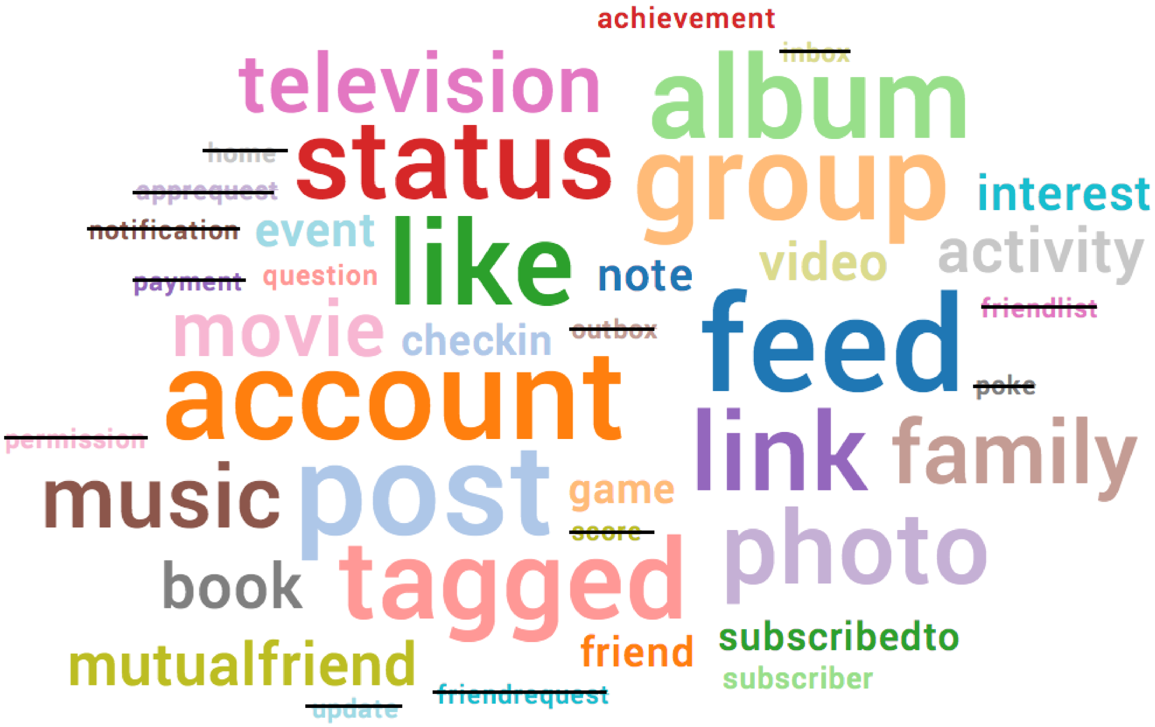


Figure 3.2: Individual visualization: Visualization of a single user's data from all categories as a snapshot in time

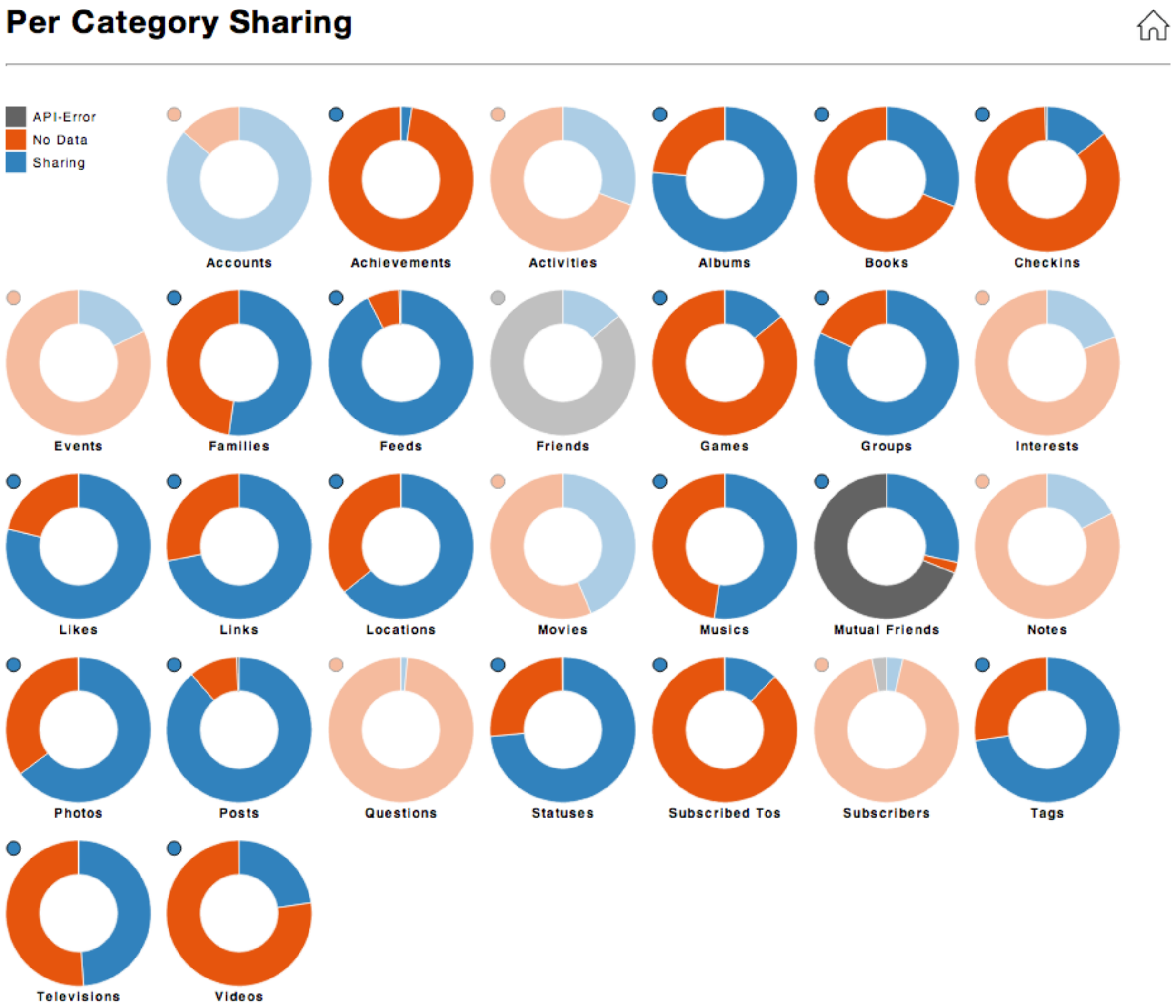


Figure 3.3: Contextualized visualization: A single user's data from as a snapshot in time (small dot top left corner) presented in the context of the same data collected from 512 users over 17 months (the larger donut)

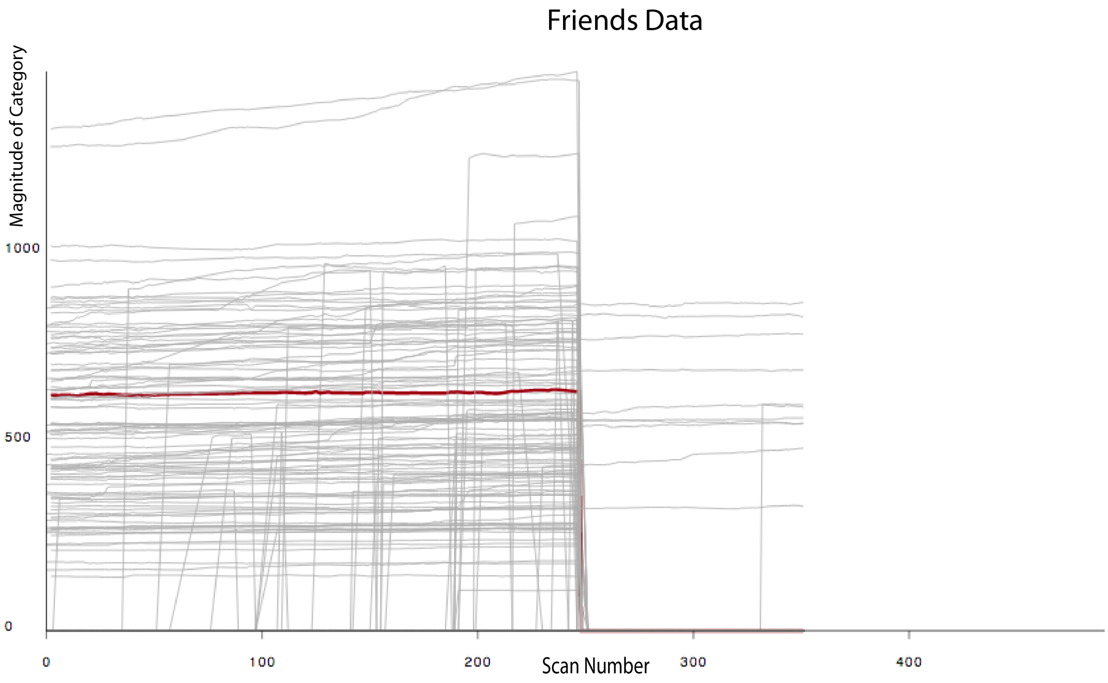


Figure 3.4: Contextualized visualization: a single user's data, for a single category presented over time, in the context of the same data collected from 512 users over a 17 month period

scale questions as well as multiple choice questions. We chose to use semantic scale questions because they allow for quantitative measurement of subjective ratings while also allowing for some flexibility in interpretation [127]. For example, one of the questions was: "Would you agree or disagree that you are concerned about online privacy?" and the answer options were: "strongly agree," "moderately agree," "neutral," "moderately disagree," and "strongly disagree." We used the 3-point scale for the majority of the semantic scale questions, as we did not need respondents to have the more granular choices as provided by the 5-point scale. It has been shown that 3-point scales do not significantly reduce reliability or validity [75]. Some of the multiple choice questions allowed respondents to enter a personalized response into a field labeled other. For example, one of the questions was: "If you would be willing to give up certain services or features in exchange for privacy, what services or features would you be willing to discontinue?" and the answer options were: "Photo Sharing",' "Seeing friends of friends," "Geotagging," "Search availability (other people can find you on the social network)," "Messaging," "Lists or Groups," and "Other." In total, the survey took 5-10 minutes to answer.

In addition to the survey, we also performed follow-up interviews with 10 survey respondents to gather additional quantitative as well as qualitative data. The follow-up interview consisted of 12 questions. For these questions, we used 3-point and 5-point semantic scales as well as yes-no questions in cases where binary responses were all that was needed. For all of these questions, we used the "'think aloud" technique, where the interviewee was told to speak freely in their response to allow us to gather deeper insights about user perspectives on privacy. Data visualizations were used to help describe the proposed privacy tools to the interviewees. The follow-up interviews were performed in person or over Skype and took from 25-45 minutes to complete depending on the extent to which the interviewee expanded on the interview question responses.

To increase reliability of the study [127], we took the following measures:

- Random order of answers: The answer options for the closed questions were randomly ordered. This ensures that the answer order does not influence the response.

- Validation questions: To ensure that respondents did not fill out the answers arbitrarily, we included two validation questions [8]. For example, one of the validation questions was: "What is the result of 5+2?" Respondents who did not answer these questions correctly were not included in the final set of valid responses.

3.3.3 Survey Respondents

We did not have any restrictions on who could fill out the survey. Because we wanted a diverse set of respondents, we distributed our survey through a variety of channels including social networks like Facebook, Twitter, and LinkedIn, various mailing lists, and to personal and professional colleagues. We also enlisted the help of colleagues to further widen the distribution of the survey.

In total 63 respondents filled out our survey between 2 August 2013 and 6 October 2013. Filtering out the incomplete and invalid responses resulted in 59 valid responses (93.6% completion rate). The survey is included in Appendix B and is also available along with summary information on our website[1].

3.3.4 Follow-up Interview Participants

After completing the survey, users were asked if they would be willing to participate in a follow-up interview. Of the 59 valid survey respondents 19 agreed to participate. From these 19, we selected 10 respondents based on schedule availability and geographic diversity. The only restriction for someone being able to participate in the follow-up interview, was that they had to be an active Facebook user. The follow-up interview included a privacy analysis that would scan the respondent's profile, and use this data to generate a visualization of their privacy setup. The visualizations that were generated are mentioned in Section 3.2.1.

3.4 Awareness and Understanding of Privacy

As mentioned earlier, one of the main goals of this study is to understand what is the current understanding of privacy settings and attitudes towards privacy for a social network user (RQ1). In this section, the findings of this study will be discussed.

3.4.1 Privacy in Social Networks

From a total of 59 users, 85% agreed that they are concerned about their privacy, while only 15% were neutral or not concerned about this issue. During the follow-up interview, it was shown that even though most of the privacy settings on Facebook are customizable with only a few clicks, most

[1] `http://www.psl.cs.columbia.edu/1494/privacy-crowdsourcing/`

from the respondents feel that this platform is "constantly changing the settings, making it hard to customize privacy". This result corroborates what others have studied [59].

Most respondents have configured their privacy settings, with 88% clearing their cookies, 85% clearing their web browser cache, and 95% deleting their web browser history. Additionally, 73% of respondents are concerned that some social networks track their online activity through their web browsing history. These numbers show that most users are concerned about their online privacy on social networks, and thus modify what information is made available to the social network.

When users were asked to rate themselves on their understanding of how to customize their privacy settings using a scale of 1 to 5, where 1 is the least understanding and 5 is the most understanding, 80% of respondents rated themselves above a 3. Using the same scale, 50% of respondents rated themselves above a 3 on their understanding of what they are and are not sharing on social networks.

However, when users were asked to change the visibility of a specific setting on Facebook (Contact Settings: Website or Contact Settings: e-mail), only half of the respondents were able to perform the task in under 3 minutes. If the respondent could not perform the task in under 3 minutes, the task was considered timed out, and there was a binary classification of "task complete" and "task incomplete." It is interesting to note the disparity between the score respondents gave themselves, as opposed to the actual knowledge they had on how to configure their privacy settings.

3.4.2 Cost of Privacy

Users were asked to answer a few questions designed to help researchers understand how much users value their online privacy. These questions refer to different cost dimensions such as monetary cost, green (ecological) cost, and service cost.

Monetary cost refers to the money people are willing to spend to protect their online privacy. Even though users from different countries were asked to answer the survey, US dollars were used as a standard metric for the survey question. A total of 22% of the respondents said they would be willing to spend money in order to have privacy in their social networks, and 13% of all respondents said they would be willing to spend \$1 - \$10 per year on privacy. For example, sites like LinkedIn have a "premium fee" or extra charge that provide users with additional privacy features in exchange for a monetary cost. It is interesting to note that although respondents are concerned about their online privacy, few are willing to pay to protect it. These results are consistent with has been researched by others [3].

Performing any operation on a computer requires energy. According to Google, serving a single user for one month emits about 8 grams of carbon per day, which is similar to driving a car for a mile [64]. We wanted to determine if users would be willing to incur in an environmental cost associated with increased privacy. From the online survey responses, 19% of the users would be willing to give up certain services in exchange for a reduced environmental cost. The services they were more likely to give up were: friends of friends visibility (15%), geotagging (12%), and individual photo sharing (8%).

When users were asked if they would be willing to give up certain services in exchange for increased privacy, 59% said they would be willing to give up some services. Some of the most common services that users were willing to give up were: geotagging (49%), friends of friends visibility (42%), and search availability (32%).

3.5 Improving User Understanding of Privacy

In order to determine if user understanding of privacy can be improved through a crowd sourced approach (RQ2), users were asked to evaluate the visualizations from Section 3.2.1. A semantic scale of 1 to 5 was used, where 1 is the least useful and 5 is the most useful. Additionally, users were asked if they would use a tool that included these visualizations.

Using the semantic scale, 70% of users rated the crowd sourced visualization tool above a 3. User appreciated the fact that they could "easily see any changes in their settings". Users were additionally asked if they would use such a tool. 80% of the respondents said they would use it at least once, arguing that the visualizations are a good way to verify if what they configured actually represents what they meant.

When asked how often would they like to be notified about their privacy settings, most respondents (90%) said they would prefer to be notified any time there is a change in their settings, 20% would like to have a monthly report, and 10% would like to have a daily report. Users were also asked how often they would like their information to be scanned (e.g., granularity for the graphs), and 50%

of respondents preferred daily granularity, 30% a monthly granularity, and 10% preferred a weekly scan.

60% of respondents would prefer to have this tool built into the social network which was accessible at any time, as opposed to 40% of the respondents that would like to be notified through email, and only 10% would like to have the service presented as a Facebook application. There were no respondents who preferred a standalone application or a browser plug-in.

3.6 Privacy Management Tools

In order to identify if there are tools for configuring privacy that are preferred over the currently provided tools (RQ3), users were presented 3 different tools for customizing their privacy settings. The tools were present in both the survey and the follow-up interview.

From the survey, 56% of the respondents said they would like to have the 3 option system for configuring privacy. 54% of respondents said they would like to have the short survey mechanism. The crowdsourcing mechanism was the least popular, with only 22% of respondents wanting to have this mechanism available for configuring their their privacy. The two popular options are mechanisms where there is no additional information about what other users are sharing, or metrics about what is more or less common to share in a particular social network.

During the follow-up interview, users were asked to rank each of these mechanisms, as well as the current mechanisms on Facebook. For this purpose, a semantic scale of 1 to 5 where 1 is the least suitable and 5 is the most suitable was used. A summary of the results is presented in Figure 3.5.

Most of the respondents, 80%, ranked the current privacy mechanisms below a 3. Users justified the low score by arguing that Facebook's current privacy mechanisms are "difficult to understand and configure". For the 3 option system, most users 50% ranked it above a 3. This system had mixed reviews. According to the respondents, one of the major benefits is that they have a "known level of privacy" which they can then later tweak to meet their specific requirements.

The survey based approach was rated either above a 3 by 70% of the participants. According to the respondents, this system's main benefit is that it doesn't have predefined standards as the first mechanism (3 option), but instead it relies on inferring the ideal settings given a user's privacy requirements. Respondents argue that some networks have similar systems in place where a tour of

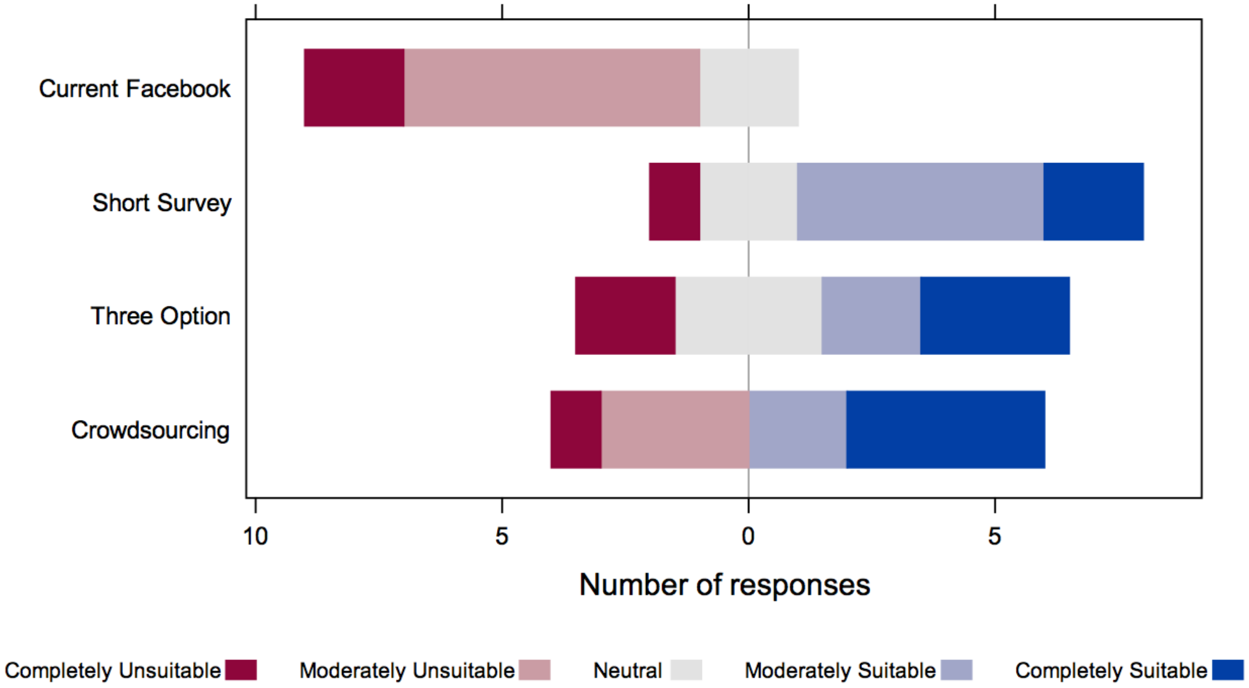


Figure 3.5: Distribution of of ratings for privacy management tools from follow-up interview

the settings is provided, yet respondents feel they don't want to "spend useful time" answering these questions, or viewing the tours in order to get a finer tuned privacy setup. However, if the survey were short enough, they would like to define their default settings using this mechanism.

The final mechanism that was presented to the respondents consisted of using crowd sourced data to provide users with additional information when configuring their privacy. This was the most polarized system, and 60% of the respondents rated this tool above a 3. Users gave positive feedback to the fact that they were able to view their privacy setup compared to another group of users information thus enabling them to mimic the settings for that particular group. However, the respondents that did not like this system claimed that "they would not trust the privacy configuration skills from other users" in the network.

It is interesting to note that, during the survey, the crowd sourced system was the least popular option, but, once users had the opportunity to ask questions and look at a mock-up of the system, it had the greatest number of respondents ranking it a 5.

3.7 Discussion

By using crowd sourced data, our approach introduces a new technique to end-users for understanding and configuring their privacy. As a first attempt, our research shows that, given the opportunity to ask questions and explore different configuration options in detail during follow-up interviews, users preferred the crowd sourced model. However, during the survey stage of research, before the subjects were able to ask questions, they rated the crowd sourced model lower than the other available options.

3.7.1 Implications of Results

Our research shows that part of the problem with current privacy controls is that users find them difficult to navigate. For example, Facebook obfuscates privacy settings by having category specific panels for certain settings instead of a single centralized set of controls. This makes it difficult for users to develop a comprehensive understanding of their individual privacy settings. When we asked users to change the website field from their Contact Settings from its current setting to some other setting (Section 3.4.2), most users immediately navigated to the general privacy settings panel which does not provide access to this field. From the 10 interviewees only half were able to successfully change the setting in the time given. This suggests having a centralized location where all privacy settings could be configured would be beneficial to users.

Some other problems with current privacy controls concern persistence, validation, and comparison. Many of the interviewees disclaimed that although they understood how they had configured their privacy, their current settings may not reflect that configuration because of how often Facebook makes changes. This shows a lack of confidence in the persistence of user configured settings and, as shown by Mashima et al. [96], this doesn't translate to user action addressing these concerns. Furthermore, this highlights the inability of users to validate how they believe their settings are configured. Finally, current settings do not provide any context to users allowing them to compare their own settings to the settings of other users. By not providing context, there is an implicit suggestion that users already know what they do and do not want to share, and that there is some inherent "correct" way to sharing that is obvious to users. Instead, we argue that users can benefit by comparing to others and our results corroborate this. Our approach addresses these problems as follows:

- **Validate:** By using crowd sourced data, our technique can provide users with an overview highlighting what information is and is not being shared.
- **Persistence:** By collecting snapshots of a user's settings over time, our approach can monitor and alert users to unanticipated changes.
- **Comparison:** The contextualized view provided with crowd sourced data allows users to quickly identify irregularities, or areas where they may find they want to adjust their settings.

Our results suggests that many users do not know about all of the categories exist in their profiles. When asked to change the website field visibility from their contact settings most users said they did not know this was a part of their profile. We believe context is especially helpful when users are configuring privacy for for categories they are unfamiliar with.

As part of the follow-up interview process, we wanted to find out not only if users preferred crowd sourced data and found it helpful, but also how they would like to be presented with the crowd sourced tools. We provided users with the following options: "Stand alone application," "Browser plug-in," "e-mail service," "Facebook app," and "Other." Almost all users said they would prefer that the tool be a built in to Facebook. This suggests getting user adoption of crowd sourced tools without the participation of social network providers would be challenging.

3.7.2 Threats to Validity

There are several limitations to the validity of our study. First, there is a selection bias because the sample of individuals that answered our survey were self-selected. This implies that our results are only applicable to the volunteering population, which does not represent a complete sample of all the social network users. Also, the sample comprises individuals from different countries and cultures, and results may vary if the study is repeated within a specific group. However, the sample is broad enough to allow us to identify statistically significant trends and relationships.

Regarding internal validity, the fact that users filled out a survey implies that only certain concerns can be identified. Also, the questions and the format in which they are presented might constrain the limits of such concerns. It is possible that the study overlooked certain concerns and preferred methodologies due to the design of the survey. However, we attempted to mitigate this validity issue by including open-ended questions where respondents could further expand their answers, and by having "think aloud" questions during the follow-up interview.

An inherent property of an online survey is that respondents may not fully understand each of the questions and they might also select arbitrary responses. To address these concerns, we included validation questions and only report statistics about respondents that correctly answered these questions.

Despite these limitations, we managed to get a diverse and large set of respondents. This increases our confidence about the overall trends that reported in this project.

3.8 Related Work

Extensive research has been focused on improving user understanding of privacy and on simplifying privacy configuration. In the following paragraphs, we highlight important examples of this research.

Many studies have focused on the divide between what information users believe they are sharing and what is actually being shared and with whom. Stutzman et al. [143] show that as Facebook users increase the ammount of information they share privately they unknowingly disclosed information to so called "silent listeners." Madjeski et al. [94] find that of the 65 participants in their study, the majority either share (94%) or hide (85%) information unintentionally. This is further supported in the findings of Liu et al. [91] showing that just 37% of users' settings coincided with their expectations and "almost always expose content to more users than expected." Johnson et al. [77] conclude that user understanding of privacy is especially at risk when considering what information is shared with "members of the friend network who dynamically become inappropriate audiences based on the context of a post." Young and Quan-Haase [169] investigate factors that influence university students to disclose personal information on Facebook. They also study the different strategies students develop to protect themselves against privacy threats. These studies all highlight the challenges confronting users when trying to manage their privacy settings on social networks but stop short of making proposals for how these challenges can be addressed. Our approach compliments the results obtained from these studies and uses this information as a basis for proposing a crowd sourced system that improves user understanding.

There are other researchers investigating and developing different tools and techniques to manage privacy in online social networks. For example, an automatic technique proposed by Fang and LeFevre [55] configures a user's privacy settings in social networking sites by creating a machine learning model that requires limited user input. Tootoonchian et al. [147] propose a system called Lockr that improves the privacy of centralized and decentralized online content sharing systems. Squicciarini et al. [142] model the problem of collaborative enforcement of privacy policies on shared data by using game theory. By extending the notion of content ownership, they propose a solution that offers automated ways to share images. Another tool, developed by Toubiana et al. [148], was designed as a geo-location aided system that allows users to declare their photo tagging preferences at the time a picture is taken. This system enforces users tagging preferences without revealing their identity. Lipford et al. [90] investigate mechanisms for socially appropriate privacy management in online social networks. They study the role of interface usability in the configuration of privacy settings and develop a first prototype where profile information is presented in an audience-oriented view. A tool called PrivAware, developed by Becker and Chen [18], detects and reports unintended information loss in online social networks. Additionally, this tool recommends action to take to mitigate privacy risk. Although these studies propose alternative tools and mechanisms for configuring privacy, to the best of our knowledge, there have been no tools that present a contextualized view of privacy during user configuration.

3.9 Conclusion

In this project, we conducted a study presenting a crowd sourced approach for simplifying the configuration and understanding of social network privacy settings. The study was divided into two sections: a survey for which we collected 59 valid responses and a follow-up interview for which 10 survey respondents participated. While only a small portion (22%) of the survey responses indicated they would prefer a crowd sourced tool for configuring privacy, during follow-up interviews 60% of participants said they would prefer such a tool over the current settings after having the opportunity to explore our approach more thoroughly.

Currently there are several obstacles that complicate user understanding and configuration of privacy. These are obfuscatory privacy control mechanisms instituted by social networks, frequent changes to privacy policy, and default settings intended to provide access to a users data. Our approach addresses these problems by simplifying comprehension of privacy through the use of data visualizations which provide an overview of the settings for each category in a user's profile.

We also provided users with data visualizations showing an individuals privacy setup and history in a contextualized view. For the individual visualizations, these are helpful as a quick summary of which categories are being shared and a quick way of determining in what ways their privacy setup has changed over time. These would allow a user to identify expected and unexpected changes in privacy by reviewing how their settings change. While some participants in the follow-up interviews did not prefer our crowd sourced approach, 70% said they found the data visualizations useful and 80% said they would use them at least once.

Our approach could be used to influence the development of future privacy control mechanisms. Although our mock-up and visualizations only serve as a template for what information such a system may include, our results show that a crowd sourced approach does improve user understanding of privacy.

Finally, as far as societal computing is concerned, our crowdsourcing technique can be leveraged to help address problems in other domains of societal computing. The basis of this approach was the large online survey described in the previous chapter. The results from that survey showed how users are equally satisfied with techniques that provide more data transparency for mitigating privacy concerns. A similar methodology can be employed to address tradeoffs in other domains of societal computing. The first step would be to gather a deeper insight from users via a survey. Based on the user feedback, new techniques could be developed and the efficacy of these new techniques can be verified using the study design described in this chapter as a template. Crowdsourcing, on its own, is a very powerful approach and we feel that there will be many different domains in societal computing where it will fit very naturally and help address many of the tradeoffs therein.

Chapter 4

Detecting Privacy Bugs via End-User Regression Testing

An added benefit of the crowdsourcing approach described in the previous chapter is that it can help detect privacy bugs in software systems. Further, we don't need access to source code to detect these bugs, i.e., they can be done from an end-user perspective. We now describe the project that focuses on this aspect of the crowdsourcing approach.

4.1 Introduction

End-user Software Engineering (EUSE) is becoming an increasingly important as "computer programming, almost as much as computer use, is becoming a widespread, pervasive practice." [83]. EUSE ranges from requirements and design to testing and debugging. End-user testing, in particular, is important for privacy because the end-users have little or no say in the functional specifications of or changes to social computing software, and because its online software they cannot avoid upgrading after each change or continue to use an "old version." Plus well-known social computing systems have an established history of making changes that breach privacy with no a priori ability for end-users to opt out [56]. But the end-users are not, in general, trained software engineers so any methodology and technology must be simple and easy to use without training.

Consider the following scenario for Pete – a user of a social system like Facebook. Pete is comfortable using websites and computers, but doesn't have a very strong technical background in

Computer Science or Software Engineering. He is worried about his privacy when he uses Facebook though. There has been a lot of media coverage about privacy concerns, how they keep changing their privacy policy periodically, how hard it is to figure out all the privacy settings, and so on and this has caused Pete some concern. Pete likes using the system to keep in touch with his friends and professional colleagues, but he doesn't want strangers to have access to his personal information, photos, likes, dislikes, etc. He has used some of the "how to" guides to configure his settings to what he wants to them (or so he thinks).

A scenario like this raises a number of interesting software engineering research challenges:

1. **R1**: Users' Mental Model of Privacy — How can we make complex privacy settings easier to understand and verify for Pete? (e.g., If I think my photos are shared with only my friends, is that really the case?) — **Requirements Engineering for Privacy.**
2. **R2**: Code/API Bugs — How can we detect if privacy settings that are in place remain the same as the software evolves and changes over time? (e.g., If my photos are currently only shared with my friends, how do I know that they won't "automatically" get shared with everyone due to a software bug?) — **Regression Testing for Privacy.**
3. **R3**: Policy Changes — How can we detect system wide policy changes that might cause privacy settings to change? (e.g., If my photos are private right now, how do we detect if there a policy change that makes all photos publicly accessible?) — **Regression Testing for Privacy.**

Ideally, for users like Pete who do not have access to the source code of systems like Facebook, we want to do this from an *end-user* perspective. In this project, we present a novel technique that leverages a social, crowdsourced approach for detecting bugs from the end-user perspective. To the best of our knowledge, this is the first technique that leverages regression testing for detecting privacy bugs.

Continuing the scenario above – consider Roger, a friend of Pete on the social system. Roger can manually monitor what part of Pete's information is visible to him. This monitoring can be done periodically as often as Pete/Roger deem necessary – say, every day, every hour, once a month, and so on.

Using this monitoring, Roger can inform Pete when the information he sees changes. For example, he might suddenly see a whole lot of new information that is now visible. This might be due to: (1)

Pete added more information manually; (2) Pete changed his privacy settings either deliberately or accidentally; (3) Pete didn't do anything – there is a bug in the code or API, possibly due to code changes; and (4) Pete didn't do anything – the social system made a wide policy change where this information for many or all of its users is now visible. Pete, now, using this feedback from Roger, can decide whether it's ok for the new information to be visible and take the appropriate actions such as changing the settings back to what he wants them to do, reviewing the privacy settings or doing nothing.

This, however, can be very tedious for Roger to have to do all this manually, particularly, if frequently done, and he could easily forget to check certain things. Thus, automated monitoring is essential and can be done if the system provides an API. A lot of social systems like Facebook [50], Twitter [152], Last.fm [87], and Google+ [63] do provide an API whose main purpose is to build an ecosystem of app developers for the system. We can leverage such APIs where possible and if an API doesn't exist, the same goal can be achieved via screen scraping.

This is our broad approach — using one's friends for detecting potential privacy violations. We call this ***Social Testing***. There are more specific details that need to be dealt with based on the platform and API and we discuss this in Sections 4.2 and 4.3. This latter section also contains details about the feasibility of this approach and the kinds of privacy bugs it has helped us uncover. The main advantages of this approach are:

1. We don't need access to source code for detecting privacy bugs. Hence, this makes it very suitable to be employed by end-users (rather than software programmers building the system).
2. It leverages the social nature of these systems for detecting these bugs.
3. It can detect privacy bugs due to changes in the code, i.e., regression testing for privacy.

The contributions of this project are:

- A novel software testing technique, called social testing, for the social circles of end-users to detect privacy bugs using regression testing. Social testing could potentially also be used for applications other than privacy preservation in social systems, such as in the multi-player gaming community;
- Two prototype tools that implement our technique for Facebook and Twitter; and

- A large empirical evaluation of our technique that demonstrates: (1) the feasibility and utility of our technique; and (2) the different kinds of bugs it can help detect.

4.2 Approach — Social Testing

The broad technique for our social testing approach is to use one's friends to help with software engineering problems. Our approach leverages the inherently social aspect of these systems, which are used for interacting and communicating with other users. This approach will apply only to systems where users are members of possibly overlapping groups and input information intended to be shared with some of these groups they are members of but not with other groups they are members of. This includes the cases of the singleton group – just me – and the universal group – everyone who uses the system, or anyone who uses the internet since many social systems often allow certain access with no login at all.

This technique could apply towards many different functional and non-functional requirements for end-users such as privacy, performance, and so on. In this project, we focus on privacy testing and in particular, on **R2** (detecting privacy bugs in code/API implementations) and **R3** (detecting system wide policy changes for privacy). There are two possible kinds of privacy violations:

- Over-sharing — From a user's point of view, this piece of information should have been private, but it can be viewed by others.
- Under-sharing — From a user's point of view, this piece of information should have been public, but it cannot be viewed by others.

We deal with both of these types of privacy violations. As our technique is intended for end-users, we assume that there is no access to source code. The main crux of our technique is — a user can choose his/her friends to periodically monitor what's visible to them via the social system. When they see a change in what's visible, this might be a privacy violation and they can inform the user. Thus, the aforementioned privacy violations, from the point of view of the tester, become: (1) Over-sharing — seeing more than you should; and (2) Under-sharing — seeing less than you should.

Thus, we use a social approach for detecting privacy bugs. The algorithm for our technique (from the tester's point of view) is outlined next: *a*) Implement/Download/Build a wrapper that can "talk" to the system under test (via an API, screenscraping, etc.). *b*) Generate a list of users to monitor. *c*) Decide on the policies (how often to monitor, which things to monitor). *d*) Based on the policies, use the wrapper to monitor the user(s). *e*) Generate diffs (i.e., differences) between the information just received and from the previous run. *f*) If there is a diff, inform the user on what changed. *g*) Repeat steps 4-6, as needed and update steps 2-3, when necessary.

We discuss the platform and API specific implementation issues, examples of privacy bugs that we found, and how this technique can help address **R2** and **R3** in the next section.

4.3 Empirical Evaluation

For our empirical evaluation, we built two prototype tools: one for Facebook; one for Twitter. Using these tools, we evaluated how our technique could help addressing **R2** and **R3**. In particular, we had two specific research questions for our empirical evaluations:

RQ1: Feasibility — Does using our technique help in detecting privacy bugs?

RQ2: Utility — What kinds of bugs does it detect? Does it help with, both, **R2** (Code/API Bugs) and **R3** (Policy Changes)?

4.3.1 Privacy and Facebook

Facebook is a great example for doing an empirical evaluation on privacy as it follows a fine-grained privacy model. It has many different privacy parameters and options; broadly, users can choose who gets to see what type of data with a lot of granularity. It provides an API, called the Graph API, that represents the Facebook social graph using objects and connections between objects [51]. Examples of the objects include User, Events, Groups, and Applications. The User object contains fields such as name, gender, and birthday and "connections" such as albums, family, groups, likes, movies, and videos. Table 4.1 shows a complete list of User Connections in Facebook and is available on the Facebook User API page [52].

Table 4.1 shows the user connections that are available via the Facebook API.

Table 4.1: User Connections as listed on the Facebook User API [52]

Name	**Description**
accounts	The Facebook apps and pages owned by the current user.
achievements	The achievements for the user.
activities	The activities listed on the user's profile.
albums	The photo albums this user has created.
apprequests	The user's outstanding requests from an app.
books	The books listed on the user's profile.
checkins	The places that the user has checked-into.
events	The events this user is attending.
family	The user's family relationships
feed	The user's wall.
friendlists	The user's friend lists.
friendrequests	The user's incoming friend requests.
friends	The user's friends.
games	Games the user has added to the Arts and Entertainment section of their profile.
groups	The Groups that the user belongs to.
home	The user's news feed.
inbox	The Threads in this user's inbox.
interests	The interests listed on the user's profile.
likes	All the pages this user has liked.
links	The user's posted links.
locations	Posts, statuses, and photos in which the user has been tagged at a location.
movies	The movies listed on the user's profile.
music	The music listed on the user's profile.
mutualfriends	The mutual friends between two users.
notes	The user's notes.

Table 4.1: (continued)

Name	Description
notifications	The notifications for the user.
outbox	The messages in this user's outbox.
payments	The Facebook Credits orders the user placed with an application.
permissions	The permissions that user has granted the application.
photos	Photos the user (or friend) is tagged in.
picture	The user's profile picture.
pokes	The user's pokes.
posts	The user's own posts.
questions	The user's questions.
scores	The current scores for the user in games.
statuses	The user's status updates.
subscribedto	People you're subscribed to.
subscribers	The user's subscribers.
tagged	Posts the user is tagged in.
television	The television listed on the user's profile.
updates	The updates in this user's inbox.
videos	The videos this user has been tagged in.

In general, there is a lot of flexibility for the user to choose the privacy settings for all of these. Users can share the data with no one, with selected friends, with all friends, with friends of friends, with certain networks (such as "Columbia University"), and with everyone. Users can also allow certain apps to access this information.

4.3.1.1 Prototype Tool

Our tool is a prototype implementation of the social technique for detecting privacy bugs in Facebook. It is easily configurable and can fetch any data provided by the Facebook API. For the purposes of this study, we focused on getting only the User Connections from [52]. The prototype tool consists of two separate components: the Data Monitor and the Diff Visualizer.

The Data Monitor is implemented as a set of Ruby scripts. It uses the Koala library [7], which is a Facebook library for Ruby and supports the Graph API. The Data Monitor works as follows: First, given a list of users, for each user, it uses Koala to get data for that user. It gets all the data listed in [52] (except the "picture" connection, which didn't exist when we started collecting data). The Facebook API supports a "Batch mode" for making data requests and we use this mode to reduce the load of the servers and to get data more efficiently. Once the Data Monitor has the data, it writes it out to a log file. We create a separate log file per user. To limit the data that needs to be stored and also for privacy reasons, we do not store the entire data; we keep only the count of data items and codify the data received according to the schema defined below:

- **0, nil** — This is used when the API returns an error. This is typically a permissions error, but can also include other server side errors.
- **1, 0** — This is used when the API returns an empty data set. This means that either there is no data in that category or that the data exists but has been hidden by the user. Note that with the latter case, it will return a 1, 0 and not a 0, nil.
- **2, x** — This is used when the API returns some data. x is the count of the number of items received. For example, if there were 10 locations that the user had been tagged at, we would log 2, 10.

The log file, thus, contains multiple lines of data. Each line contains two things: (1) The current timestamp when the data was received; and (2) An array containing the username and the 41 arrays for the connections encoded into the schema shown above. A sample log file is shown in Listing 4.1.

The Diff Visualizer, which is also a set of Ruby scripts, parses each log file and creates a human readable output if there is a diff (i.e., difference) between any consecutive runs for a user. If there is a difference, it will print out the pairwise timestamps and what the old and the new values are. We divide a difference into two categories: a Major Difference and a Minor Difference. A Major Difference occurs when, for a certain connection, the data received changes the codifying categories. For example, if music was 2, 8 and became 1, 0, this would be a Major Difference. A Minor Difference, on the other hand, occurs when the data changes, but does not change the codifying category. For

```
Fri Apr 27 13:12:30 −0400 2012, ["someUser", [2, 259], [2, 1], [1, 0], [1, 0], [2,
    25], [1, 0], [1, 0], [1, 0], [2, 19], [1, 0], [2, 3], [2, 25], [1, 0], [1, 0],
    [1, 0], [2, 25], [2, 20], [2, 25], [2, 4], [1, 0], [2, 1], [1, 0], [1, 0], [0,
    nil], [0, nil], [1, 0], [1, 0], [0, nil], [0, nil], [2, 1], [0, nil], [0, nil],
    [0, nil], [0, nil], [1, 0], [0, nil], [1, 0], [1, 0], [1, 0], [1, 0], [0, nil]]
Tue May 01 15:22:35 −0400 2012, ["someUser", [2, 259], [2, 1], [1, 0], [1, 0], [2,
    25], [1, 0], [1, 0], [1, 0], [2, 18], [1, 0], [2, 3], [2, 25], [1, 0], [1, 0],
    [1, 0], [2, 25], [2, 20], [2, 25], [2, 4], [1, 0], [2, 1], [1, 0], [1, 0], [0,
    nil], [0, nil], [1, 0], [1, 0], [0, nil], [0, nil], [2, 1], [0, nil], [0, nil],
    [0, nil], [0, nil], [1, 0], [0, nil], [1, 0], [1, 0], [1, 0], [1, 0], [0, nil]]
```

Listing 4.1: Sample Log File for someUser. Note: We anonymized the username and replaced it with someUser. The order of arrays is as follows: friends, accounts, apprequests, activities, albums, books, checkins, events, feed, interests, likes, links, movies, music, notes, photos, posts, statuses, tagged, television, videos, achievements, family, friendlists, friendrequests, games, groups, home, inbox, locations, mutualfriends, notifications, outbox, payments, permissions, pokes, questions, scores, subscribedto, subscribers, updates.

example, if music was 2, 10 and became 2, 12, this would be a Minor Difference. We, thus, have two variants of the Diff Visualizer, which will print out either Major or Minor Differences as needed. A sample output containing both Major and Minor Differences is shown in Listing 4.2.

4.3.1.2 Feasibility

The first step in our empirical evaluation was to show the feasibility of our approach and tool. For this step, we used the tool to access the Facebook data of a research colleague of the first author. Facebook uses OAuth 2.0 [71], which is an open standard for authentication. We provided our tool with the first author's OAuth 2.0 access token so that the tool can access the same data that the first author can. This is the equivalent of the research colleague, using our social approach, asking the first author to monitor his information on Facebook. For this step of the evaluation, we did the following:

1. We accessed the Facebook data of the research colleague (name anonymized, for privacy reasons).

```
======May 25 2012 and May 27 2012======
friends - Old: 2, 783, New: 2, 784
events - Old: 2, 1, New: 1, 0
feed - Old: 2, 15, New: 2, 16
likes - Old: 2, 202, New: 2, 200
posts - Old: 2, 6, New: 2, 7
tagged - Old: 2, 9, New: 2, 10
======May 27 2012 and May 28 2012======
friends - Old: 2, 784, New: 2, 783
posts - Old: 2, 7, New: 2, 8
tagged - Old: 2, 10, New: 2, 9
```

Listing 4.2: Sample Diff Output for a user. events is an example of a Major Difference; the others are all Minor Differences.

2. After the data was accessed, we asked the colleague to turn on privacy controls and make the data less visible. This would enable us to check if our tool could detect changes in privacy, where data is made less visible. The colleague did this by adding the first author to one of his pre-defined friend lists that had very limited access to his profile. We accessed the data using our tool again.
3. Finally, we asked the colleague to turn off the privacy controls and make the data more visible. This would enable us to check if our tool could detect changes in privacy, where data is made more visible. We accessed the data again using our tool.

The output from the Diff Visualizer is shown in Figure 4.1. As the figure shows, turning on the privacy settings reduces visibility — things like photos, locations, and feed were visible earlier and the Facebook API responses contained data; with the privacy settings on, the Facebook API returns an empty set. Turning off the privacy settings makes the data visible again, as seen in the right hand side of the figure.

Our Facebook prototype tool can thus detect changes in privacy settings. These changes can either be data being made more private or data being made less private. Thus, if someone suddenly starts sharing more or less data than before, our tool would detect this and this could indicate a privacy bug. Next, we show some examples of bugs our tool can help detect.

```
=====Apr 24 2012 and Apr 25 2012=====
feed - Old: 2, 19, New: 1, 0
photos - Old: 2, 25, New: 1, 0
posts - Old: 2, 19, New: 1, 0
tagged - Old: 2, 5, New: 1, 0
videos - Old: 2, 1, New: 1, 0
locations - Old: 2, 1, New: 1, 0
```

(a) Turning On the Privacy Settings

```
=====Apr 25 2012 and Apr 27 2012=====
feed - Old: 1, 0, New: 2, 19
photos - Old: 1, 0, New: 2, 25
posts - Old: 1, 0, New: 2, 20
tagged - Old: 1, 0, New: 2, 4
videos - Old: 1, 0, New: 2, 1
locations - Old: 1, 0, New: 2, 1
```

(b) Turning Off the Privacy Settings

Figure 4.1: Changing Privacy Settings — Output as seen from our tool

4.3.1.3 Facebook Bugs — Family and Friendlists

We ran our Facebook prototype tool using the first author's access token and collected data for all his Facebook friends ($n = 516$). The data was collected roughly every day for each user for approximately eleven weeks (from May 1, 2012 to July 20, 2012). For each user, the number of data points (i.e., the number of days on which we successfully got data from the Facebook servers) was, on average, 36.24 ($\sigma = 3.26$, median = 36, max = 44, min = 25). (Please see Section 4.4 for a discussion on the number of data points and on the robustness of our approach.)

Upon running the Diff Visualizer, we found that 63.18% (326 out of 519) of the users had Major and Minor Differences during the data monitoring period. Out of these, there were a total of 5065 Minor Differences (on average, 15.54 per user) and 780 Major Differences (on average, 2.39 per user). Not all of these differences necessarily imply privacy bugs; some of these differences would arise from the "normal" use of these systems, i.e., users adding new photos from a recent trip and so on. But even in these cases, the users may not be aware with whom they are now sharing this new information.

We now highlight, in this and the next subsection, a couple of interesting case studies on the bugs that were discovered using our tool. For a certain user, there was one family member being shown for the first three weeks of the data monitoring period. On May 23, 2012, our tool found a lot of Major Differences (there had been other, unrelated, Minor Differences previously) for that user. The output from our tool for the entire monitoring period is shown in Listing 4.3. The last diff shows a lot less data being visible. Was this a case of a user turning on privacy settings? Or

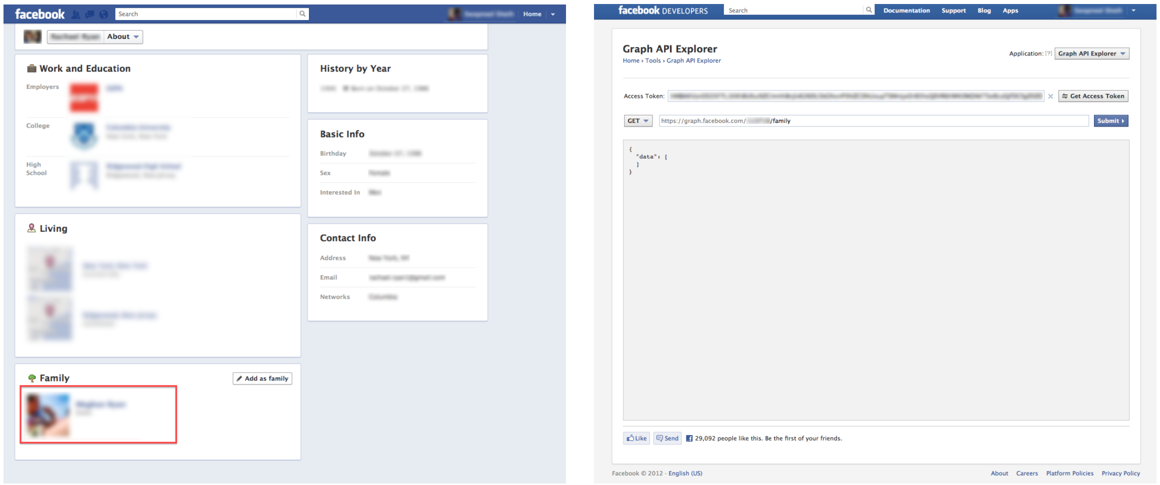
```
======May 04 2012 and May 07 2012======
feed - Old: 2, 21, New: 2, 20
posts - Old: 2, 15, New: 2, 14
======May 11 2012 and May 14 2012======
feed - Old: 2, 20, New: 2, 19
tagged - Old: 2, 11, New: 2, 10
======May 18 2012 and May 21 2012======
posts - Old: 2, 14, New: 2, 15
tagged - Old: 2, 10, New: 2, 8
======May 22 2012 and May 23 2012======
albums - Old: 2, 3, New: 1, 0
feed - Old: 2, 19, New: 1, 0
likes - Old: 2, 2, New: 1, 0
photos - Old: 2, 25, New: 1, 0
posts - Old: 2, 15, New: 1, 0
tagged - Old: 2, 8, New: 1, 0
family - Old: 2, 1, New: 1, 0
groups - Old: 2, 2, New: 1, 0
locations - Old: 2, 4, New: 1, 0
```

Listing 4.3: Facebook Family Bug — Output as seen from our tool

perhaps did something change in the Facebook Code or API resulting in a bug? We tried to find out the cause. We were, of course, limited in our efforts as we don't have access to the source code. We decided to focus on the family connection, which lists a user's family members. This is highlighted in red in Listing 4.3.

The first step of our investigation was to see the actual page on Facebook. On the page, the family member was still visible. This is shown in Figure 4.2a, highlighted in red (user details have been blurred out to protect privacy). To ensure that there were no bugs in our tool implementation, we used the Facebook Graph API Explorer and verified that this returned an empty data set. This is shown in Figure 4.2b.

Finally, we started looking at Facebook bug reports to see if anyone else had reported a similar issue. We found two relevant bug reports. The most relevant bug report was titled "Can no longer access FriendList members on test users" [53]. In this bug report, a user had created two test users



Access Token: [redacted]

GET https://graph.facebook.com/[redacted]/family

```
{
  "data": [
  ]
}
```

(a) The Facebook Website — The highlighted red rectangle shows the family member.

(b) API — Using the API, the list of family members is empty.

Figure 4.2: Facebook Family Bug — On the website, you can see the family member; Using the API, you cannot see the family member.

and added each user to the other's family list. When the user tried accessing the members of one of the test user's family, the Graph API returned an empty data set. This is exactly the same behavior as what we observed here. Facebook have confirmed that the bug exists and assigned it to a developer with medium priority. The second bug report, titled "Some of the friendlists do not show members from Graph API, why??" [54], reported a similar problem to the previous bug report. In this bug report, the user had many friend lists, which is a generalization of the family connection. Upon accessing the data from the Graph API, some of the friend lists return all the members; some return only a subset of the members. Facebook has responded to this bug report by triaging it with low priority.

These bug reports and our tool output, taken together, lead us to believe that there is a bug in the Facebook API, that was recently introduced due to code changes and should not have passed the regression testing phase. This is an example of under-sharing — less information is public than it should be. Our tool was able to detect this privacy bug using end-user regression testing.

```
======Jul 02 2012 and Jul 06 2012======
feed - Old: 2, 14, New: 2, 15
links - Old: 1, 0, New: 2, 23
posts - Old: 2, 3, New: 2, 5
tagged - Old: 2, 14, New: 2, 15
```

Listing 4.4: Facebook Links Bug — Output as seen from our tool

4.3.1.4 Facebook Bugs — Links made public

At the start of July, our tool discovered another example of a privacy bug — this time, it was an example of over-sharing. The output from our tool for one user is shown in Listing 4.4. There was a lot of data made public about links posted by a user, which is highlighted in red. The interesting part was that this was not data that was recently added by the user; some of these links were added back in 2009. This was data that had been on Facebook for a while and not visible to friends of the users; now, it was visible.

Analyzing the data further, we found that 73 out of 516 users were now sharing a lot more links than before and that this new data was first visible towards the start of July. Of the 73 users, 57 (i.e., 11.05%) were sharing more than one link, thus indicating that this was not new information added by the user recently.

To understand the cause behind this over-sharing, we conducted an informal open-ended interview with one of the users, Keith (name changed for privacy reasons). Excerpts from the interview are shown next (reproduced with permission from Keith). On being told that he is now sharing 23 links, which weren't visible earlier, Keith responded: "whoa [. . .] ok...... that is weird." After looking at the links that were visible, Keith said: "ok, all of these links are valid, but, am surprised you can see them [. . .] I, as a developer, opened my account for developer access. it's the only way possible and I just thought I was authorizing that one app. they must have their permissions f***ed up [. . .] it's either that, or facebook changed my settings automatically. " Keith said that he would change his settings back so that the links would not visible anymore and ended the interview with: "good thing your app [sic] was able to catch it."

Keith is currently a student at Columbia University pursuing a Ph.D. in Computer Science. Given that someone with a technical background didn't know about his data being public and wasn't completely sure what changed, a *non-technical end-user* would generally have a much harder time figuring out changes in privacy settings. This is the main strength of our tool — giving end-users an easy way for detecting potential privacy breaches. Since this particular bug had not affected all the users, it seems to indicate that this a Policy Change that is being rolled out gradually by Facebook. Alternatively, this could be another example of a bug (affecting only some users) in the Facebook Code or API. Either way, our tool was able to detect this.

4.3.2 Privacy Testing and Twitter

Twitter, as opposed to Facebook, has a completely different model with respect to privacy. While Facebook has a very fine-grained control model for controlling what's visible to whom, Twitter has a very coarse-grained model. Users can choose if their accounts are "protected" or not, with the account being not protected as the default setting [153]. If the account is not protected, all tweets are public and can be viewed by anyone. If the account is protected, it can only be viewed by the followers of that person. There is no mechanism for deciding this on a per-tweet basis, for example. Regardless of whether the account is protected or not, anyone can still see the number of tweets, the number of followers, and the number of following for any user.

Thus, the only setting that matters in terms of privacy is whether the account if protected or not. Our Twitter tool, described below, uses the same social approach for detecting if the privacy settings change. In addition to this, to show the generalizability of our approach in dealing with different kinds of social systems, we decided to treat some other user information as "sensitive" — i.e., if Twitter had more fine-grained controls for these types of information, our approach (and tool) would still be able to detect changes in privacy settings. A partial list of user fields from the Twitter User API is shown in Table 4.2. For the purposes of our tool and empirical study, we treated these as sensitive information as well and inform the user when these change.

4.3.2.1 Prototype Tool

Our prototype tool can detect privacy bugs for Twitter. Similar to the Facebook prototype tool described in Section 4.3.1.1, it is easily configurable and can fetch any data provided by the Twitter API. For the purposes of this study, we focused on getting only the partial list of User fields shown

Field	**Description**
description	The user-defined UTF-8 string describing their account.
favourites_count	The number of tweets this user has favorited in the account's lifetime.
followers_count	The number of followers this account currently has.
friends_count	The number of users this account is following (AKA their "followings").
geo_enabled	When true, indicates that the user has enabled the possibility of geotagging their Tweets.
listed_count	The number of public lists that this user is a member of.
protected	When true, indicates that this user has chosen to protect their Tweets.
statuses_count	The number of tweets (including retweets) issued by the user.
verified	When true, indicates that the user has a verified account.
withheld_in_countries	When present, indicates a textual representation of the two-letter country codes this user is withheld from.
withheld_scope	When present, indicates whether the content being withheld is the "status" or a "user."

Table 4.2: Partial list of Users Fields from the Twitter User API [155]

in Table 4.2. Similar to the Facebook tool, the Twitter tool consists of two components: the Data Monitor and the Diff Visualizer.

The Data Monitor is implemented as a set of Ruby scripts. It uses the Twitter library [109] to get data from the Twitter API. The Diff Visualizer, similar the Facebook Diff Visualizer, is also a set of Ruby scripts that parses each log file and creates a human readable output if there is a diff between any consecutive runs for a user. The rest of the workings of the Data Monitor are similar to the Facebook tool described in Section 4.3.1.1. We do not repeat the implementation details of the tools due to space limitations. The main difference is that this tool focuses on the user fields shown in Table 4.2; the rest of the implementation is similar. The other difference is that, for Twitter due to its lack of fine-grained privacy controls, we do not distinguish between Major and Minor Differences.

```
========2012-06-19 and 2012-06-19========
friends_count-Old: 2, 140, New: 2, 142
========2012-06-19 and 2012-06-19========
protected-Old: 2, false, New: 2, true
========2012-06-19 and 2012-06-19========
protected-Old: 2, true, New: 2, false
========2012-06-21 and 2012-06-22========
followers_count-Old: 2, 138, New: 2, 137
statuses_count-Old: 2, 548, New: 2, 551
```

Listing 4.5: Privacy Monitoring of @swapneel — Output as seen from our tool

4.3.2.2 Feasibility

We ran our Twitter prototype tool and collected data for some of the first author's research colleagues ($n = 10$). The data was collected roughly every day for each user for approximately four weeks (from May 19, 2012 to July 20, 2012).

We also collected data for the first author (@swapneel) and changed the account to protected (and back to open) and verified if the tool can detect changes in privacy settings. The tool could, indeed, pick up the changes in privacy settings. The output from the Diff Visualizer (highlighted in red) in shown in Listing 4.5.

One of the twitter accounts for which the data was collected was the official ICSE twitter account (@ICSEconf). If we assume that the fields listed in Table 4.2 are sensitive information, our tool can detect changes in these as well. The partial output from the Diff Visualizer is shown in Listing 4.6.

Recently, a bug was found in Twitter where users who wanted to follow others were "arbitrarily, randomly, and haphazardly" unfollowed [114]. This "unfollow" bug was acknowledged by the Twitter team and they said that they were working on a fix. Our tool would have been able to detect this bug as well as follows: Say I started following two new users today. If the output from the Diff Visualizer was anything other than two, we know that there is a bug with following someone. The user could then check which user got unfollowed and follow the user again, if needed.

```
========2012−06−19 and 2012−06−20========
statuses_count−Old: 2, 904, New: 2, 906
========2012−06−20 and 2012−06−21========
listed_count−Old: 2, 67, New: 2, 68
statuses_count−Old: 2, 906, New: 2, 910
========2012−06−21 and 2012−06−22========
followers_count−Old: 2, 877, New: 2, 876
statuses_count−Old: 2, 910, New: 2, 912
========2012−06−22 and 2012−06−23========
followers_count−Old: 2, 876, New: 2, 878
statuses_count−Old: 2, 912, New: 2, 913
========2012−06−23 and 2012−06−25========
followers_count−Old: 2, 878, New: 2, 879
========2012−06−25 and 2012−06−26========
followers_count−Old: 2, 879, New: 2, 880
========2012−06−26 and 2012−06−27========
followers_count−Old: 2, 880, New: 2, 882
========2012−06−27 and 2012−06−28========
followers_count−Old: 2, 882, New: 2, 883
========2012−06−28 and 2012−06−29========
followers_count−Old: 2, 883, New: 2, 885
friends_count−Old: 2, 1015, New: 2, 1016
```

Listing 4.6: Privacy Monitoring of @ICSEConf — Output as seen from our tool

4.4 Discussion

4.4.1 Flexibility

One advantage of this approach for detecting privacy bugs is the flexibility. A user could choose different sets of friends to monitor different things, if the social system has a fine-grained privacy model. For example, he could have a friend check the privacy settings of his photos and check-in locations. He could have someone from his network (such as “New York”) check his music and movies. He could have a friend of a friend check his feed. He could also create overlapping groups — his friends should be able to see albums and locations; his network can only see the albums. Thus, he could use different sets of friends to verify that the privacy settings indeed are what he expects them to be and to alert him when they can see more or less than what they saw before.

A user can also choose how often the data is fetched based on how active he is on the social system, the API rate limits, and personal preferences such as the tradeoff between the load on the social system's servers and his privacy needs.

Finally, in terms of implementation, our prototype tools were stand-alone tools that ran off the command line. Social systems like Facebook and Twitter provide rich ecosystems for apps. Our approach can be implemented as apps that run on Facebook, for example. Other alternatives include a browser plugin that automatically runs the regression testing when the user logs into one of these systems or on a periodic basis (every hour, every day, and so on), a "normal" desktop application with a GUI, and so on. We used Ruby for our implementations; this was of out choice and is not a constraint. Any programming language that can access the web can be used. Having a wrapper library for that programing language does help as it obviates the need to deal with lower level protocol details. For example, Twitter has a list of libraries for 14 programming languages ranging from Java, .NET, and Python to Erlang, Scala, and Clojure [154]. There are no inherent implementation or UI limitations as far as our approach is concerned.

4.4.2 Generalizability

Another advantage of this approach for detecting privacy bugs is the generalizability of the technique. In general, it can work with any social system regardless of what kinds of privacy controls and features it has. The previous Section showed how it could work with systems at two extreme ends of the privacy spectrum: Facebook, with its fine grained settings for choosing who sees what and Twitter with its coarse grained settings, which is essentially an on/off switch.

Having an API to use makes it much easier to implement a tool for a particular social system. This, however, is not a limiting factor — if there is no API, the same approach can be combined with alternatives such as screen or web scraping.

4.4.3 Robustness

Our approach is also very robust — it does not need that the Data Monitor is run every day or that the Data Monitor successfully fetches data for each user every day (or on every run). In spite of the Facebook API having slow response times [119] and timing out occasionally, which resulted in our Data Monitor fetching data successfully on average 36 times (out of possibly about 75 times) for each user in an eleven week time period, our tool still works fine and detects bugs as shown in the previous Section. The tradeoff with fetching data less often is that we will not be able to catch transient bugs. For example, if something is made public only for a few minutes and then it is private again, if our Data Monitor is not active then, we will not be able to detect it.

In contrast to the Facebook API, the Twitter API, in our experience, was much more stable. Regardless of the stability and reliability of the social system under test, as mentioned above, our approach can still detect bugs due to its highly flexible nature.

4.4.4 Limitations and Threats to Validity

A limitation of our prototype tools is that we keep track of the number of items in data fetched, rather than the actual data. For example, in the Facebook tool, for a user, we log that the user had ten photos rather than what the photos are. We do this for two reasons: (1) to reduce the data that needs to be stored; and (2) for privacy reasons. Due to this, our tools might miss out on changes in privacy if a user, for example, deletes one photo and adds one photo, our tool would see this as no change having occurred. This, however, is a limitation of our *prototype tools*, and not of our *approach*. If an end-user wishes to keep track of the exact data, rather than the number of data items, our tools can be modified to do that. An added challenge, in this case, would be to find the semantic similarities between data items, which would be easier in some cases (e.g., checkin locations, groups) than others (e.g., albums, interests, likes).

An inherent limitation of our approach is the possibility of false negatives, i.e., privacy bugs that exist in the system that our tool/approach is not able to catch. There might be bugs that have existed since the first version of the software; if there is no change in the code, our regression testing approach will not work. There might be privacy bugs that affect only some of the users; if the users that are currently using our tool are not affected by this bug, we won't be able to detect it. The main reason for this is that our approach is intended for *end-users* and we don't have access to the source code of the system under test. From an *end-user* perspective, it's hard to detect bugs using our approach that may not have an external manifestation or change in behavior for our users.

Finally, coming back to our software engineering research questions, even though **R1** is beyond the scope of this project, we make a couple of observations based on our empirical results. If there are never any changes in the social software, then **R2** and **R3** won't happen, but in a limited indirect way just trying to use our tool (which will keep saying "no change") might make users more aware of privacy settings issues and thus in a very small way help with **R1**. Now if there *are* changes, so **R2** and/or **R3** come into play, then the awareness with respect to **R1** would be stronger because users would then be prompted to go look more closely at the particular settings that were affected and thus would understand them better and adjust their mental model accordingly.

Statistical Conclusion — Do we have sufficient data to make our claims? For our Facebook tool, we fetched data for 516 users resulting in almost 18,700 data points. For our Twitter tool, we fetched data for 10 users resulting in almost 350 data points. The goal of the empirical evaluations was to find examples of privacy bugs to show the feasibility and utility of our approach, which we did find as described in Section 4.3.

External Validity — Do our results generalize to other systems? Our prototype tools were implemented for two different systems: Facebook and Twitter. Our approach is broad and can apply to any social system as discussed in Sections 4.4.1 and 4.4.2 above.

4.5 Related Work

Many recent studies on online social networks show that there is a (typically, large) discrepancy between users' intentions for what their privacy settings should be versus what they actually are [77, 91, 94]. For example, Madejski et al. report that, in their study on Facebook, 94% of their participants ($n = 65$) were sharing something they intended to hide and 85% were hiding something that they intended to share. Liu et al. [91] found that Facebook's users' privacy settings match their expectations only 37% of the time. This is **R1** mentioned earlier.

In addition to the problem of understanding existing privacy settings, there are two orthogonal problems. First, there might be software bugs in the implementation of the privacy settings, which results in over-sharing or under-sharing of information, and as software evolves over time, this might introduce new bugs. This is **R2** mentioned earlier.

Second, systems like Facebook change their policies on privacy often and these changes in policy usually end up confusing users even more. Dan Fletcher [56] writes: "In the past, when Facebook changed its privacy controls, it tended to automatically set users' preferences to maximum exposure and then put the onus on us to go in and dial them back. In December, the company set the defaults for a lot of user information so that everyone — even non-Facebook members — could see such details as status updates and lists of friends and interests. Many of us scrambled for cover, restricting who gets to see what on our profile pages." This is **R3** mentioned earlier and these are the main research problems that we are trying to solve with our approach.

There have been some recent papers on data privacy and software testing. Clause and Orso [33] propose techniques for the automated anonymization of field data for software testing. They extend the work done by Castro et al. [32] using novel concepts of path condition relaxation and breakable input conditions resulting in improving the effectiveness of input anonymization. Taneja et al. [145] and Grechanik et al. [65] propose using k-anonymity [144] for privacy by selectively anonymizing certain attributes of a database for software testing. These papers propose novel approaches using static analysis for selecting which attributes to anonymize so that test coverage remains high. Peters and Menzies [120] propose an anonymization technique for sensitive data so that it can be used for cross-company defect prediction. They show that it is possible to make data less sensitive and still maintain high utility for data mining applications.

Our work is orthogonal to these papers on data anonymization. The problem they address is — how can one anonymize sensitive information before sharing it with others (e.g., sending it to the teams or companies that build the software, sharing information for testing purposes, and sharing data across multiple companies, respectively)? The problem we address is - how can *end-users* verify if the software systems they are using are handling privacy correctly? Further, all these papers are trying to protect the privacy of the data. We, on the other hand, are trying to detect privacy violations and test if the systems have any privacy bugs.

There has been a lot of work in the field of regression testing mainly towards test case selection and test case prioritization [72, 76, 123, 172], including a very detailed, and excellent, recent survey by Yoo and Harman [168] and the references therein. Our work builds on, and differs from, all of the above in two aspects – our regression testing approach is targeted towards *end-users* and is targeted towards finding *privacy* bugs.

There has also been some recent work in using taint analysis for detecting security and privacy violations [46, 133]. These approaches require access to source code for taint analysis. Our approach, on the other hand, is targeted towards end-users who do not have source code access to the social systems that they are using. Our social testing approach is similar in some ways to "do you see what I see," a technique proposed in the networking community to support distributed fault detection and diagnosis from the client-side [137], although there the actual end-users are not directly involved, and is also related to the network security community's collaborative intrusion detection, e.g., [115], where the goal is to share data about penetration attempts against different organizations' enterprise networks but without inadvertently sharing any private information.

4.6 Conclusion

Privacy in social systems is becoming a major concern. End-users of such systems are finding it increasingly harder to understand the complex privacy settings. Even if they do understand the settings, as the software evolves over time, there might be bugs introduced that breach users' privacy. There might also be system wide policy changes that could change users' settings to be more or less private than before.

We present a novel technique, called ***Social Testing***, that can be used by *end-users*, as opposed to software developers building the system, for detecting changes in privacy, i.e., regression testing for privacy. This technique can broadly apply towards functional and non-functional requirements for end-users such as privacy, performance, and so on. In this project, we applied our technique towards detecting privacy bugs from an end-user perspective. Broadly, a user can use his/her friends to monitor what information is visible in the social systems and to automatically detect when more or less information is visible, thus indicating a potential privacy concern. We also presented two prototype tools — one for Facebook ; one for Twitter — that implemented our technique for detecting privacy bugs. The results of our evaluation show the utility and feasibility of our approach and tools for detecting privacy bugs. Our Facebook tool discovered that 63.18% of the users had differences in privacy where they were sharing either more or less information than before.

In particular, we focused on two case studies of bugs that we found and upon interviewing one user affected by the bug, the user said: '[. . .] am surprised you can see them (new information that was recently made visible, which was detected by our tool) [. . .] good thing your app was able to catch it" and that he would change his settings back to what they should have been. To the best of our knowledge, this is the first technique that leverages regression testing for detecting privacy bugs from an end-user perspective.

Finally, as far as societal computing is concerned, end-user testing of software systems will become increasingly important. As software systems increase in complexity and scope and there are limited resources, software systems are (and will continue) being deployed that have bugs in them. Especially in the context of societal computing and the tradeoffs that exist, these bugs might be more sensitive and crucial to end-users. This is where our Social Testing approach can come to the fore. Our approach can be applied to a variety of domains and used by a diverse set of end-users to continue testing the deployed systems. This would be similar to "perpetual testing" [113]. It might also help reduce the testing load on central servers and, perhaps, better balance it among the end-users of these systems.

Chapter 5

Money for Nothing, Privacy for Free

Privacy has been an important topic for research in recent times and has been studied by many different communities. In most of these cases though, dealing with privacy typically requires additional computation and CPU overhead. We feel that with oil reserves running low, global warming, and the energy crisis balancing privacy and CPU overhead will become an important concern in the future. We have discovered a technique where it is possible to get privacy "for free", i.e., without any extra overhead as far as CPU computation is concerned. We now describe this project; we call it "Money for Nothing, Privacy for Free."

5.1 Introduction

In this project, we propose an approach, which we call "Privacy for Free," targeted towards online social systems. In particular, we focus on systems that already have access to user data such as purchase history, movie ratings, music preferences, and friends and groups and that use complex data mining techniques for providing additional social benefits such as recommendations, top-n statistics, and so on to their users. In the software engineering community, these are systems like Mylyn [81], Codebook [19], or others([57,149]) that have access to user (developer or end-user) interactions with software artifacts such as code, bug reports, and test cases. The problem we deal with is users who have intentionally disclosed data on a public system, entering their data via web browsers onto some website server that is known to make publicly available certain data-mined community knowledge gleaned from aggregating that data with other users — but the users don't want their data to be personally identifiable from the aggregate.

The main research question we try to answer here is — Is there an approach that can be used with complex web applications and software systems, that will achieve privacy without spending any extra resources on computational overhead? We believe it is — our key insight is that we can achieve privacy as an accidental and beneficial side effect of doing already existing computation.

The already existing computation in our case is weighting user data in a certain way — weighting recent user data exponentially more than older data to address the problem of "concept drift" [164] — to increase the relevance of the recommendations or data mining. This weighting is very common and used in a lot of systems [39, 74, 84, 105]. Recent work in the databases/cs theory communities on Differential Privacy [43, 98] led to our insight that our already existing computation for weighting user data is very similar to one of the techniques for achieving differential privacy. Intuitively, differential privacy ensures that a user's participation (versus not participating) in a database doesn't affect his privacy significantly. We provide more detailed information on Differential Privacy in following subsections. This resulted in the formulation of our hypothesis — if we change the concept drift computation so it matches the technique for achieving differential privacy (which would be a very minor and straightforward code change as the two techniques are very similar), would we get privacy as a beneficial side effect of addressing a completely different problem?

We show that it is indeed possible to get privacy as a beneficial side effect of addressing concept drift — thus, privacy for free — and this is the main contribution of this project. Our approach can be used in certain social computing systems and web applications to achieve "privacy for free," and we show the feasibility, sustainability, and utility of using this approach to building software systems. We also contribute to the discussion in the privacy community about how to define privacy and how to achieve it. Specifically, we suggest a new direction for designing (differentially, or otherwise) private algorithms and systems motivated by using the beneficial side-effects of doing some already existing computation.

We now provide some background information on Differential Privacy and Concept Drift.

5.1.1 Differential Privacy

In the 1970s, when research into statistical databases was popular, Dalenius [38] proposed a desideratum for statistical database privacy — access to a statistical database should not enable someone to learn something about an individual that cannot be learned without access to the database. While such a desideratum would be great for privacy, Dwork et al. [41, 43] showed that this notion of absolute privacy is impossible using a strong mathematical proof. The problem with the desideratum is the presence of "Auxiliary Information". Auxiliary Information is similar to, and a generalization of, the notion of Correlation Privacy mentioned earlier.

Dwork gives a nice example to explain how Auxiliary Information can be a problem when privacy is concerned — "Suppose one's exact height were considered a highly sensitive piece of information, and that revealing the exact height of an individual were a privacy breach. Assume that the database yields the average heights of women of different nationalities. An adversary who has access to the statistical database and the auxiliary information "Terry Gross is two inches shorter than the average Lithuanian woman" learns Terry Gross' height, while anyone learning only the auxiliary information, without access to the average heights, learns relatively little." An interesting observation made by Dwork is that the above example for breach of privacy holds regardless of whether Terry Gross' information is part of the database or not.

To combat Auxiliary Information, Dwork proposes a new notion of privacy called Differential Privacy. Dwork's paper is a culmination of the work started earlier and described in papers such as [23, 40, 42]. Intuitively, Differential Privacy guarantees privacy by saying that if an individual participates in the database, there is no additional loss of privacy (within a small factor) versus if he had not participated in the database. Formally, Differential Privacy is defined as follows: A Randomized function K gives ϵ-differential privacy if for all data sets D_1 and D_2 differing on at most one element, and all $S \subseteq Range(K)$,

$$\Pr[K(D_1) \in S] \leq \exp(\epsilon) \times \Pr[K(D_2) \in S] \tag{5.1}$$

The notion of all data sets D_1 and D_2 captures the concept of an individual's information being present in the database or not. If the above equation holds, it implies that if an individual's information is present in the database, the breach of privacy will be almost the same if that individual's information was not present. Differential Privacy is now commonly used in the database, cryptography, and cs theory communities [24, 41, 44, 128].

We like the definition of Differential Privacy due to its strong mathematical foundations, which can allow us to prove/disprove things theoretically. From a software web application developer's point of view, they can tell their users — "Look, our system is differentially private. So if you decide to use our system and give it access to your data, you are not losing any additional privacy (within a small factor) versus if you did not use our system. In other words, the probability of bad things happening to you (in terms of privacy) is roughly the same whether you use our system or not."

5.1.2 Achieving Differential Privacy

Dwork describes a way of achieving differential privacy by adding random noise. In the Terry Gross height example above, instead of giving the true average, the system would output average$\pm\delta$, where δ would be randomly chosen from a mathematical distribution. Thus, the adversary wouldn't be able to find out the exact height of Terry Gross. Since then, there have been many papers that have proposed different mechanisms for achieving differential privacy [24, 41, 44, 128].

A mechanism of note for achieving differential privacy was proposed by McSherry and Talwar [98] called the "Exponential Mechanism" (EM). The EM algorithm is as follows: Given a set of inputs, and some scoring function that we are trying to maximize, the algorithm chooses a particular input to be included in the output with probability proportional to the exponential raised to the score of the input using a scoring function. Thus, inputs that have a high score from the scoring function have an exponentially higher probability of being included in the output than those inputs that have a low score. McSherry and Talwar prove that this EM algorithm is differentially private.

Consider the Terry Gross example from above and let's assume that the database has historical data going back 100 years. The average heights of people change over time so giving an average height over the 100 years is not very useful. If the scoring function we use is to maximize the recency of data, newer data elements will be chosen with exponentially higher probability that older data elements to be included in the average. Since we are doing this probabilistically, the exponential probability weighting ensures that the exact answer is not revealed and that differential privacy is maintained. This EM algorithm is one of the corner stones of our "Privacy for Free" approach and we describe how it's used in the next section.

5.1.3 Concept Drift

People's preferences change over time — things that I like doing today may not be things I liked doing 10 years ago. If data is being mined or recommendations being generated, the age of the data needs to be accounted for. To address this problem, the notion of Concept Drift was formed [164]. This problem needs to be addressed by any field that deals with data spanning some time frame (from a few hours to months and years). An example class of systems that need to address the problem on Concept Drift is Recommender Systems. Many recommender systems use Collaborative Filtering (CF), i.e., recommending things to an individual by looking at what other users similar to the individual like [73, 105, 171]. CF algorithms typically look at the activities of individuals from the past (movies watched, things bought, etc.) and use this to derive recommendations. However, people's preferences change over time. For example, when I am in college and taking a lot of classes, I might buy a lot of textbooks from Amazon. When I graduate, I may not need textbook recommendations. This is exactly the kind of problem that Concept Drift tries to address.

Other example classes of systems that need to address this problem are social software systems [19], systems for collaboration and awareness [163], systems that mine online software repositories [31], etc. For these kinds of systems, there is a lot of old and recent data available and weighting certain data differently might be essential.

5.1.4 Addressing Concept Drift

There have been many different solutions proposed to address the problem of Concept Drift [82,85,164]. A particular solution of note is the Exponential Time Decay Algorithm [34] (ETDA, henceforth). ETDA weights things done recently exponentially higher than things done in the past. It gradually decays the weight of things done in the past so that things done in the distant past do not affect the outcome as much as things done recently, thus addressing the problem on Concept Drift.

$$g(x) = \exp(-l \times x), \text{ for some } l > 0 \tag{5.2}$$

The non-increasing decay function using by ETDA is shown in Equation (5.2). ETDA is very popular and used by a lot of systems [39, 74, 84, 105]. For the rest of the thesis, we refer to this as the CD (Concept Drift) algorithm.

```java
public double getWeightedValue() {
  double value = 0;
  for(int i=0; i<array.length; i++) {
    double weight = Math.exp(-i);
    value += weight*array[i];
  }
  return value;
}
```

Listing 5.1: Java code for the CD algorithm

```java
public double getWeightedValue(double epsilon) {
  double value = 0;
  for (int i = 0; i < array.length; i++) {
    double weight = Math.exp(-i * epsilon);
    double probability = Math.random();
    if (probability < weight) {
      value = array[i];
      return value;
    }
  }
  return value;
}
```

Listing 5.2: Java code for the EM algorithm

Consider the Terry Gross example again and let's assume that the database has historical data going back 100 years. As average heights change over time, the CD algorithm will weight newer data exponentially higher than older data resulting in a weighted average height. This would reflect the recent trends but also account for older data. The CD algorithm is the another corner stone of our approach and we build on it more in the next section.

5.2 Approach — Privacy for Free

The EM algorithm can use a variety of scoring functions — McSherry and Talwar show different scoring functions for privacy preserving auctions [98]. In such scenarios, the EM and CD algorithms are not similar. Using the timestamp scoring function is what makes them similar to each other. The CD algorithm uses exponential weighting over the data while the EM algorithm chooses inputs with probability proportional to the exponential of the scoring function.

Only if we choose the scoring function for the EM algorithm to be the timestamp of the data, the two algorithms becomes similar. The CD algorithm is deterministic and weights new data exponentially higher than older data; the EM algorithm is probabilistic and chooses new data with an exponentially higher probability than older data.

The Java code for the CD algorithm and the EM algorithm using the timestamp scoring function are shown in Listings 5.1 and 5.2 respectively. In terms of running time complexity, the CD algorithm is $O(n)$. For EM (using the timestamps scoring function), the worst case is also $O(n)$. However, as we use randomization, the expected running time is sublinear — $o(n)$.

This is the crux of this project — if existing systems that already use the CD algorithm modify the code to use the EM algorithm instead, they would, as an added benefit, get the main advantage of the EM algorithm — differential privacy. Further, this privacy would not require any extra computational overhead and thus, we would get privacy for free.

Since these two algorithms are very similar, it would require a very small and straightforward change to the code to change from the CD algorithm to the EM algorithm. We would need to replace the CD code with the EM code shown above. This would be a one-time change and could be done by adding a new library method for EM or done statically via refactoring and could even be automated.

The important requirement for the differential privacy guarantees to hold are that all the data access must be done via the EM algorithm, which could be implemented as a separate class or be part of a library or the data model, etc. A standard way to implement this would be to use the Decorator or Adapter design pattern for data access.

5.3 Evaluation

Our approach requires implementing (or substituting an existing implementation of the CD algorithm with) the EM algorithm. To evaluate our approach, we implemented the EM and CD algorithms and investigated the differences in these. Our goal was to answer the following research questions:

RQ1: Feasibility—Does using our approach guarantee differential privacy?

RQ2: Utility—Does using our approach affect the utility of the system to give meaningful recommendations or mine data?

RQ3: Sustainability—Can our approach be sustainable? Can using our approach result in no additional computational resources for privacy?

With RQ1, we aim to prove the primary benefit of our approach — guaranteeing privacy. Our goal is to show that it does indeed guarantee differential privacy making it suitable to be used in a variety of social systems and web applications.

With RQ2, we explore the utility of using our approach. A “straw man” way to guarantee privacy for any recommender/data mining system is to give a random answer every time. This would not require any clever technical solutions, but this would be very bad for the overall utility of the system — the goal of most such systems is to provide relevant information. There exists a tradeoff between accuracy and privacy and we explore this here. We aim to show that, using our technique, there is a small loss in accuracy and that this loss in accuracy scales very well (roughly constant) as the size of the system increases. Thus, if a small loss in accuracy is acceptable, we can get privacy for free without spending any additional computational resources.

With RQ3, we aim to show the sustainability benefits of using our approach. We show that using our approach (and the EM algorithm) requires less CPU time than the equivalent CD algorithm. Not only do we not need any additional computational resources, we should be able to reduce computational needs by using our approach.

5.3.1 RQ1 — Feasibility

Our approach requires the use of the EM algorithm for all access to the data. The EM algorithm that we require is exactly the same as the one proposed by McSherry and Talwar [98]. The algorithm they propose can work with different scoring functions that weight the data differently — in our case, the scoring function we use is the timestamp of the data. Our use of the EM algorithm in our approach can thus be viewed as an instantiation of the general EM algorithm. McSherry and Talwar show a theoretical proof for the EM algorithm to be differentially private. We do not repeat the proof here and we encourage the interested reader to look at the paper (page 5 of [98]). As all data access happens via the EM algorithm, our approach also guarantees differential privacy.

5.3.2 Methodology — Synthetic Data

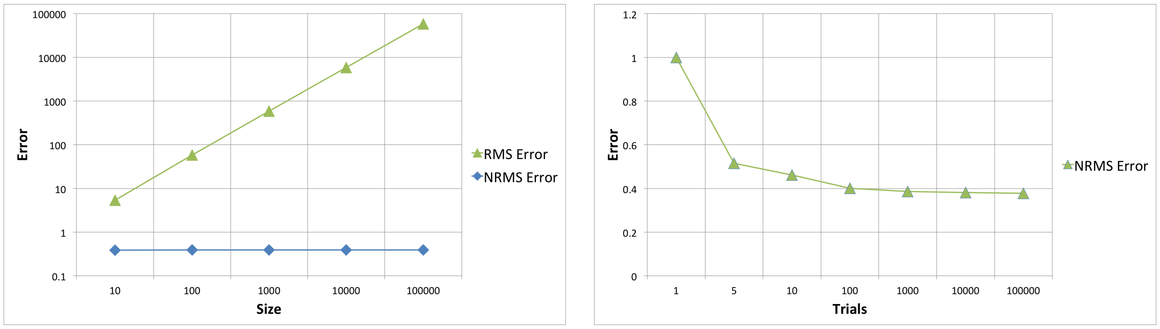
For RQ2 and RQ3, we carried out experiments to validate our hypotheses. We use synthetic data for the first set of experiments. For experiments with real world data, please see Section 5.3.5. We create an array of size n and randomly fill it with values from 0 to $n-1$. Each element has a timestamp associated with it to simulate user activity — for the purpose of this experiment, we assume that the timestamp is the array index. A lower array index indicates that the item is newer. Thus, we want to prefer items with a lower index in the output as these items indicate things that are done recently.

Using the differential privacy EM algorithm [98], we choose the scoring function to be maximized by returning a value with as low an array index as possible. Thus, we choose elements from the array with probability based on their array index. We set $epsilon = 0.5$.

In the experiments, we randomly generate the array and compute the score using the CD and the EM algorithms. We then plot the RMS and normalized RMS errors between these two algorithms. The error is the difference in the score returned by the CD and the EM algorithm. The CD algorithm will give us the "true" score; the EM algorithm (as it tries to preserve privacy) will give us a close approximation. We discuss the results in the following subsections.

5.3.3 RQ2 — Utility

For the first set of experiments, we varied the size of the array and plotted the RMS and normalized RMS errors between the CD and EM algorithms. The results are shown in Figure 5.1a. To smooth out the noise in the experimental results (as CD is a deterministic algorithm while EM is a probabilistic one), we ran the experiment 1000 times with each array size and took averages. The graph shows us that as the size of the input array increases, the RMS error increases linearly — this is expected as with larger array sizes, the entries in the array have correspondingly larger values (due to our



(a) RMS and NRMS Error vs. Size of data set

(b) NRMS Error vs. Number of Trials

Figure 5.1: Utility and error using synthetic data

methodology), resulting in linearly increasing RMS error. Meanwhile, the normalized RMS error is roughly constant.

This shows us the tradeoff between accuracy and privacy. We observe that in these experiments, the loss of accuracy is relatively small — the normalized RMS error is less than 0.4. Thus, irrespective of the data set size, switching to the EM Algorithm (as required by our approach) from the CD Algorithm will not worsen the accuracy of the algorithm by more than the constant factor, and we have the added benefit that the EM algorithm also guarantees differential privacy. Whether the loss of accuracy is acceptable or not (or a worthy price to pay for the free privacy) is subjective and we deliberately do not enter a philosophical debate here (is accuracy of the system more "important" than user privacy? who decides this? the user? the web application developers?). Many papers in the database and theory communities have explored the tradeoffs between privacy and accuracy (e.g., [23, 40, 97, 98]) — our key point in this section is that yes, there is a loss of accuracy, but no worse than accepted in [97]. A limitation of our approach is that if this loss of accuracy is not acceptable for certain systems, our approach will not work.

For our second set of experiments, we varied the number of trials keeping the size of the array fixed to 1000. As the value computed using the EM algorithm is probabilistic in nature, we carry out multiple runs (called trials here) and take the average value over all the trials to smooth out the value. The graph plotting the NRMS error vs. the number of trials is shown in Figure 5.1b. This graph shows us that as the number of trials increases, the NRMS error reduces. Thus, initially, even though there may be a bigger error between the CD and EM algorithms, in the long run, the error will be small (but not zero, as a zero error would imply returning the accurate answer and thus, not preserving privacy).

With these set of experiments, we explored the utility of our approach. For an existing system (that may already use an algorithm similar to the CD one), a one-time change would be required to add in the EM algorithm and retrofit the system to our approach. This change is relatively straightforward and could even be automated. Making such a change, albeit results in a small loss of accuracy, gives the huge benefit of getting privacy for free without spending any additional computational resources.

5.3.4 RQ3 — Sustainability

For RQ3, we want to show the sustainability of our approach. With the EM algorithm in place, what we ideally want is that our system does not take any additional computational resources. We decided to use the CPU processing time to estimate the computational resources needed by the two algorithms. We instrumented the CD and EM algorithms and measured how long they took in the first set of experiments in Section 5.3.3 above. The resultant graph is shown in Figure 5.2. The graph shows us that for all data sizes the EM algorithm took less CPU time than the CD algorithm.

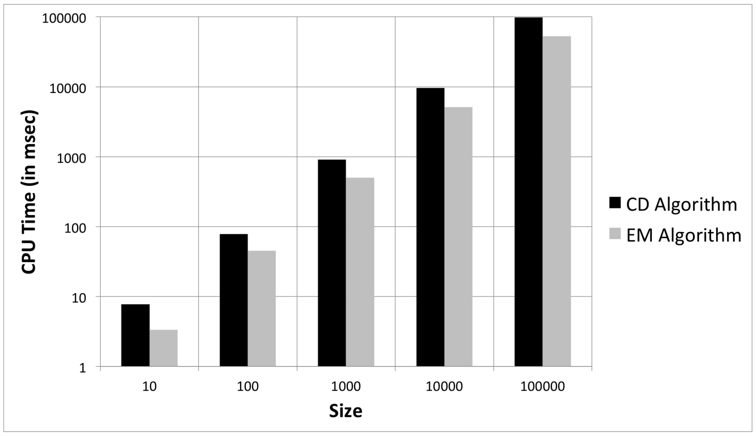


Figure 5.2: Sustainability benefits of replacing the CD algorithm with the EM algorithm

Not only does the EM algorithm not require any additional computational resources, it actually reduces the existing computation. Thus, changing to our approach will make the software system even more sustainable.

5.3.5 RQ2 Redux — Utility using real world data

For the next set of experiments, we evaluated our approach (in particular, RQ2 Utility) using real world data, viz., the MovieLens dataset [68]. This dataset, provided by the GroupLens research project, contains numerous data sets from the MovieLens web site, which consist of movie ratings by a large number of users. These datasets are commonly used in various recommender system evaluations [45, 58, 69, 80, 117, 173]. For our experiments, we used the "1 Million" data set from MovieLens data set, which contains 1,000,209 anonymous movie ratings of approximately 3900 movies by 6040 users.

We used Apache Mahout, which is a large open-source machine learning and recommender system library [10]. We implemented our CD and EM algorithms in Java. We used the Decorator design pattern for the implementation and this can serve as a guideline for other system designers who want to integrate the EM algorithm into their own systems with minimal changes. We leveraged the functionality of Mahout for user-based collaborative filtering as follows: For each user in the data set, we generated top 10 movie recommendations using various combinations of User Similarity and User Neighborhoods. User Similarity is used to denote how "similar" two users are and common metrics for these are Cosine Similarity, Pearson Correlation, Spearman Correlation, and so on. User Neighborhoods are a key part of user recommender systems as they create "neighborhoods" of similar users and these are used for generating recommendations. Neighborhoods are typically computed using metrics such as k-Nearest Neighbors (for different values of k) and Threshold Neighbors (all users that are above a certain threshold of similarity are considered neighbors.).

For the experimental setup, we used k-Nearest Neighbors (for $k = 3$ and $k = 10$). The user similarity metrics chosen were Pearson and Spearman Correlation. The values of epsilon ϵ, the differential privacy parameter, were 0.02, 0.5, 1, and 2. For each combination, we first generated the "default" recommendations. Next, we generated recommendations using our CD and EM algorithms by weighting the original data as described in the sections above.

Using the default recommendations, we calculated precision and recall for the 10 recommendations that were generated for each user. The results are shown in Figure 5.3. The Y-axis for all the graphs shows precision and recall and the X-axis is epsilon. The graphs show that as epsilon increases, the precision and recall for each setting of the collaborative filtering algorithms decreases.

Default Recommendations	CD Recommendations	EM Recommendations
Shawshank Redemption, The (1994, Drama)	Rocky (1976, Action, Drama)	**Stand by Me (1986, Adventure, Comedy, Drama)**
Dead Poets Society (1989, Drama)	Saving Private Ryan (1998, Action, Drama, War)	**Dead Poets Society (1989, Drama)**
Stand by Me (1986, Adventure, Comedy, Drama)	**Shawshank Redemption, The (1994, Drama)**	**Shawshank Redemption, The (1994, Drama)**
Green Mile, The (1999, Drama, Thriller)	Odd Couple, The (1968, Comedy)	Life Is Beautiful (La Vita e' bella) (1997, Comedy, Drama)
Like Water for Chocolate (Como agua para chocolate) (1992, Drama, Romance)	Bridge on the River Kwai, The (1957, Drama, War)	**Like Water for Chocolate (Como agua para chocolate) (1992, Drama, Romance)**
Silence of the Lambs, The (1991, Drama, Thriller)	**Jurassic Park (1993, Action, Adventure, Sci-Fi)**	**Silence of the Lambs, The (1991, Drama, Thriller)**
Jurassic Park (1993, Action, Adventure, Sci-Fi)	Exorcist, The (1973, Horror)	**Green Mile, The (1999, Drama, Thriller)**
Alien: Resurrection (1997, Action, Horror, Sci-Fi)	—	To Kill a Mockingbird (1962, Drama)
Die Hard 2 (1990, Action, Thriller)	—	Shine (1996, Drama, Romance)
Platoon (1986, Drama, War)	—	Gone with the Wind (1939, Drama, Romance, War)

Table 5.1: Comparison of top-k recommendations using the three techniques (default, CD, EM) for a typical user. We used Pearson Correlation and k-Nearest Neighbor for $k = 3$. The movies in bold are the ones recommended in more than one technique.

Table 5.1 shows an example of the qualitative difference between the recommendation algorithms. For this experiment, we chose $\epsilon = 0.02$, Pearson Correlation and $k = 3$ using k-Nearest Neighbor. We show the top-10 movie recommendations for a typical user. The movies highlighted in bold are the ones that are in common for at least 2 of the recommendation algorithms. Note that the CD algorithm only generated seven recommendations for this user. The results show us that there is some overlap between the default algorithm and CD algorithm; on the other hand, there is a lot of overlap between the default algorithm and the EM algorithm. As shown in the quantitative results above, the amount of overlap for the EM algorithm can be configured based on the differential privacy parameter ϵ.

There are two implications for system designers from our experimental results. First, if they really like the recommendation results from the CD algorithm, they can choose an epsilon of their choice and get the same precision and recall as before. Other metrics in recommendation algorithms might be important, but that is beyond the scope of this work. System designers could create an experimental study similar the one in this section and use it as a benchmark. Second, the different privacy parameter epsilon gives system designers a lot of flexibility for fine-tuning their individual systems. There is a tradeoff between accuracy and privacy and epsilon can be chosen based on various stakeholder interests.

5.3.6 Threats to Validity

The notion of Differential Privacy may not relate to the user-centric view of Privacy as users might think it "strange" that the system assumes that bad things can happen anyway — the guarantee it gives is just regarding whether the user data is part of the system or not. While that is true, we feel that differential privacy has many compelling arguments in its favor — the biggest, for us, is not having to decide what data is sensitive and what is not. The differential privacy algorithms treat all data as sensitive making it easier not to leak data by accident. One would, therefore, not have to deal with the subjective nature of deciding what's sensitive. We also feel that the guarantee might actually make it even more compelling for the user. From their point of view — "participating is not going to make my privacy any significantly worse.. so I might as well participate."

Finally, this work doesn't help in scenarios of non-temporal data access. We used the IMDB/Netflix examples earlier to make the general problem familiar to the reader; we address a special case of the problem where timestamps are available. In the differential privacy area, it's proven that for *any* method that has any utility, there exists side information that will break privacy on individual records. With differential privacy approaches such as the EM algorithm, the guarantees that exist for each individual are that participating in the database will not add to the risks that are already there. Finally, in the scenario of non-temporal data access, the use of the concept drift would not be applicable either.

5.4 Related Work

Privacy has become an increasingly important topic for the community at large. A lot of different research communities are looking at the impact of privacy and techniques for improving privacy for users. Some examples of these communities are sociologists, computer scientists, HCI, etc. We discuss some of the relevant related work next.

Fang and LeFevre [55] proposed an automated technique for configuring a user's privacy settings in online social networking sites. Paul et al. [118] present using a color coding scheme for making privacy settings more usable. Squicciarini, Shehab, and Paci [142] propose a game-theoretic approach for collaborative sharing and control of images in a social network. Toubiana et al. [148] present a system that automatically applies users' privacy settings for photo tagging. All these papers propose new techniques that are targeted to making privacy settings "better" (i.e., more usable, more visible) from a user's perspective. Our approach, on the other hand, targets the internal algorithms such as recommendations used by these systems.

There have been some recent papers on data privacy and software testing. Clause and Orso [33] propose techniques for the automated anonymization of field data for software testing. They extend the work done by Castro et al. [32] using novel concepts of path condition relaxation and breakable input conditions resulting in improving the effectiveness of input anonymization. Our work is orthogonal to the papers on input anonymization. The problem they address is — how can users anonymize sensitive information before sending it to the teams or companies that build the software? The problem we address is — how can systems that already have access to user data (such as purchase history, movie preferences, and so on) be engineered so that they don't leak sensitive information while doing data mining on the data? Further, the aim of our approach is to provide privacy "for free," i.e., without spending extra computational resources on privacy. The input anonymization approaches require spending extra computation (between 2.5 minutes to 9 minutes) as they address a different problem. We believe that the our approach can be combined with the input anonymization approach if needed. If users are worried about developers at the company finding out sensitive information, input anonymization is essential. If, however, they are worried about accidental data leakage through the data mining of their information, using the "Privacy for Free" approach may be more suitable. This would also make the software system more sustainable as we don't spend any computation doing the anonymization of the inputs.

Taneja et al. [145] and Grechanik et al. [65] propose using k-anonymity [144] for privacy by selectively anonymizing certain attributes of a database for software testing. Their papers propose novel approaches using static analysis for selecting which attributes to anonymize so that test coverage remains high. Similar to above, our approach is orthogonal as we focus on an approach that will prevent accidental leakage of sensitive information via data mining or similar techniques. Further, these approaches using k-anonymity also require significant additional computational resources and thus, may not be sustainable when energy resources are scarce.

The testing problem above is concerned with internal data that users keep on their own computers and do not want to disclose outside their own computer (or put into a server and the testing is on that server software, but the data was understood to be specific to that user and never aggregated with other users). The problem we deal with instead is users who have intentionally disclosed data on a public system, entering their data via web browsers onto some website server that is known to make publicly available certain data-mined community knowledge gleaned from aggregating that data with other users — but the users don't want their data to be personally identifiable from the aggregate.

Finally, work on input anonymization and k-anonymization both focus on software testing whereas our approach focuses on an approach for building privacy preserving systems or re-engineering existing software systems with minimal code changes (since only the parts affected need to be changed) with a specific goal — to make privacy sustainable and not require additional resources.

There has also been a lot of work related to data anonymization and building accurate data models for statistical use (e.g., [5,49,88,121,159]). These techniques aim to preserve certain properties of the data (e.g., statistical properties like average) so they can be useful in data mining while trying to preserve privacy of individual records. Similar to these, there are has also been work on anonymizing social networks [21] and anonymizing user profiles for personalized web search [175]. The broad approaches include aggregating data to a higher level of granularity or adding noise and random perturbations. As we are interested in sustainable ways of achieving privacy, these approaches are not applicable as they typically require (a lot of) extra computational effort.

While there has been a lot of interest (and research) in data anonymization, we would like to reiterate that only data anonymization might not be enough. Narayanan and Shmatikov [107] demonstrate a relatively straightforward way of breaking the anonymity of data. They show how it is possible to correlate public IMDb data with private anonymized Netflix movie rating data resulting in the potential identification of the anonymized individuals. Backstrom et al. [12] also describe a series of attacks for de-anonymizing social networks that have been anonymized to be made available to the public. They describe two categories of attacks — active attacks where an evil adversary targets an arbitrary set of users and passive attacks where existing users try to discover their location in the network and thereby cause de-anonymization. Their results show that, with high probability and modest computational requirements, de-anonymization is possible for a real world social network (in their case, LiveJournal [26]). Finally, Zheleva and Getoor [174] show it's possible to infer private profiles of users on social networks based on their groups and friends.

5.5 Discussion

The crux of this project, and the novel idea, is that it is possible to combine two existing approaches to *increase* the degree of privacy in social computing systems, under certain conditions. This poses an interesting open problem — Are there other algorithms that we currently use for solving some problem that also accidentally provide privacy or some other added benefit?

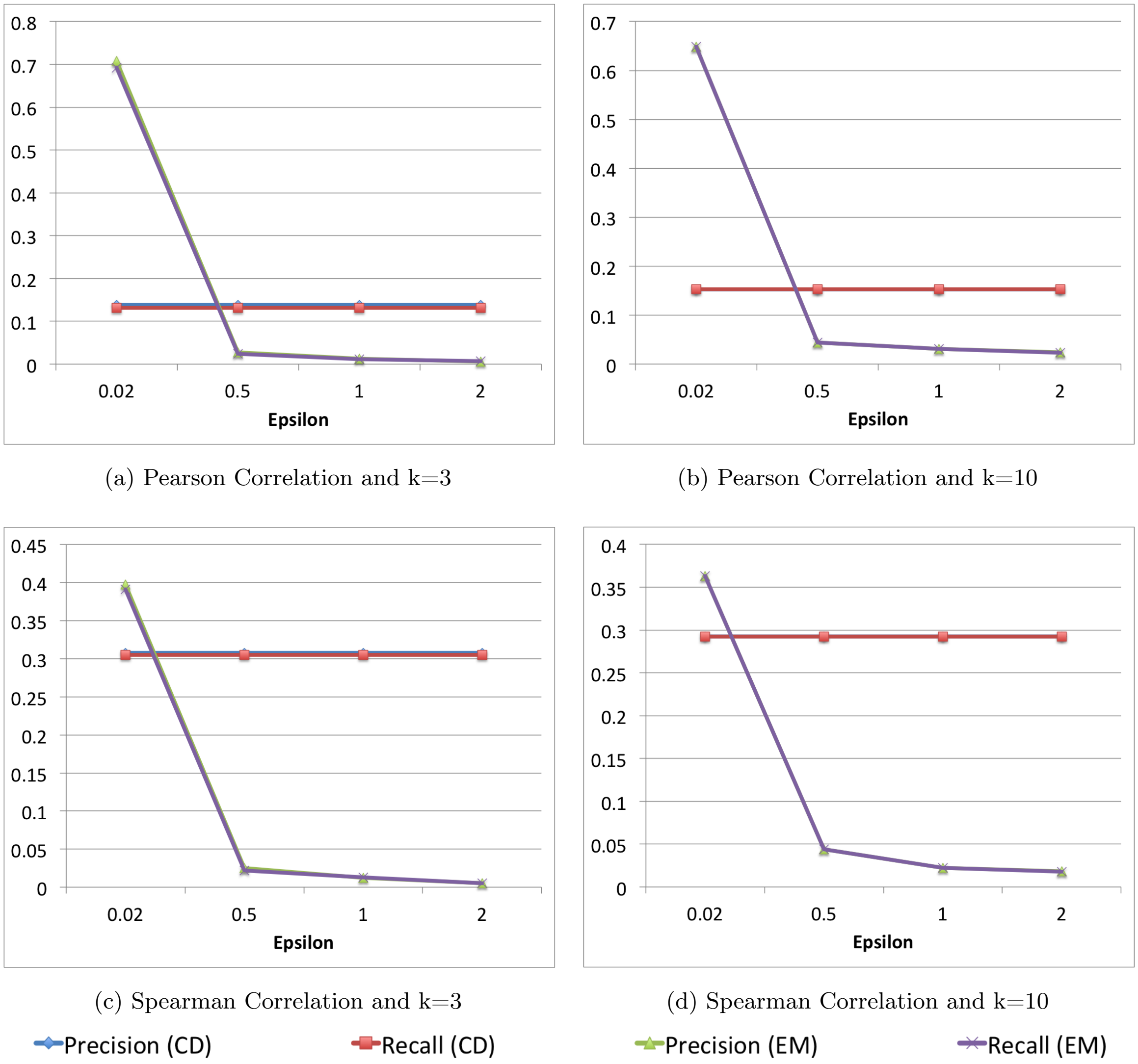
A lot of research in the theory and cryptography community on differential privacy has focused on Mechanism Design [24, 41, 44, 128]. Mechanism design is the process of coming up with new mechanisms that are differentially private and solve certain problems in domains such as machine learning and statistics. The previous sections hint at an interesting avenue of future research — Mechanism Discovery. We discovered how the CD algorithm as a side effect may provide differential privacy for free. It might be fruitful to look at currently used algorithms in varying domains and see if they too, as a side effect, provide differential privacy. This might lead to the discovery of generalized mechanisms for differential privacy that can be used in other domains, which have not yet been proposed or discovered by theory and cryptography researchers. Mechanism discovery might act as a great complement to the Mechanism Design research.

In order for mechanism discovery to be successful, a greater emphasis must be placed on multidisciplinary research. Even though there is some research in recommender system privacy [20, 136], most of the papers do not use a formal and precise definition of privacy. Our community could benefit a lot from the precise and formal use of differential privacy. Similarly, most of the theory and cryptography community may not be aware of the privacy research done by our, or other, communities. There might be a lot of interesting discoveries of mechanisms suitable for differential privacy. The only way any of this can be achieved is by a greater emphasis on multidisciplinary research using areas such as systems, theory, cryptography, web, and databases.

5.6 Conclusion

As social computing systems that collect users' data proliferate, privacy has and will continue to become a major concern for the society at large. The main research question that we wanted to answer is — Is there an approach that can be used with a certain web applications and software systems, that will achieve privacy without spending any extra resources on computational overhead? Our "Privacy for Free" approach can achieve privacy as an accidental and beneficial side effect of addressing concept drift. The results of our evaluations show the feasibility, utility, and in particular, the sustainability of our approach as it does not require any additional computational resources to guarantee privacy.

Finally, as far as societal computing is concerned, this project shines a new light on addressing many tradeoffs in different domains. We saw how it was possible to get privacy as a beneficial side-effect of doing some already existing computation. Discovering such side-effects in other systems might help reduce the strain on some of the tradeoffs. However, such discovery relies heavily on multidisciplinary research. Since a lot of time and research effort is currently being spent on mechanism design (as described in the section above), we advocate an increased emphasis of mechanism discovery, i.e., looking at other domains and verifying if those approaches can help here as well. Further, our approach of combining two known techniques can also serve as a blueprint for future tradeoffs in societal computing.



(a) Pearson Correlation and k=3

(b) Pearson Correlation and k=10

(c) Spearman Correlation and k=3

(d) Spearman Correlation and k=10

Figure 5.3: Utility of the CD and EM algorithms using real world data

The Y-axis for all the graphs show precision and recall.

Chapter 6

Introduction to Societal Computing

In the previous chapter, we saw how it was possible to get added privacy without adversely affecting CPU computation time. This, however, is just one example of a tradeoff that we might have to make going forwards. There are other "grand challenges" facing society right now and in the near future [29, 108]. Many of these challenges will involve building complex software systems as part of addressing these challenges. They will likely involve many tradeoffs, which need to be taken into account especially as far as software engineering is concerned. In this chapter, we generalize some of these tradeoffs from earlier and call it "Societal Computing". We also describe how this thesis can help with addressing some of these tradeoffs by reusing various parts of it and applying it to other domains.

Today's college students do not remember when social recommendations, such as those provided by Amazon, Netflix, Last.fm, and StumbleUpon, were not commonplace. The rise of Web 2.0 and social networking has popularized social computing as a research area. Established research communities such as Human Factors [158], Computer Supported Cooperative Work, and Software Engineering have fostered emerging topics such as Recommender Systems [61, 89, 170] and Social Software Engineering [19, 70, 105, 130, 131]. However, social computing is primarily concerned with achieving individual benefits from community participation, and not so much with addressing the societal downsides – in particular that those individual benefits may come at community expense or even the longer-term expense of the individual.

We present a novel problem – or perhaps a novel way of looking at known problems – that we believe has not yet been explored by the community. Thus we propose and define "**Societal**

Computing," a new research area for computer scientists in general and software engineering and programming language communities (SE/PL, hereafter) in particular, concerned with the impact of computational tradeoffs on societal issues. Societal Computing research will focus on aspects of computer science that address significant issues and concerns facing the society as a whole such as Privacy, Climate Change, Green Computing, Sustainability, and Cultural Differences. In particular, Societal Computing research will focus on the research challenges that arise due to the tradeoffs among these areas. An example of such a tradeoff could be a complex software system that needs to comply with varying laws in different regions or countries. While complying with such laws is important for the society as it might safeguard the interests of individuals, doing so might require investing considerable computer resources through the entire software lifecycle, which might not be a good idea when Green Computing is concerned. As complying with laws would be mandatory, the option should not be to ignore the law but perhaps to lobby to change the law, by making regulators aware of the green computing implications, or perhaps choose not offer this software (or maybe just turn off the affected features) in locales that retain the expensive laws. Most of these are legal/business decisions and not an SE/PL concern, but the SE/PL community can make it easier to orthogonalize features (and thus make simple to turn off without breaking everything else) whose compliance with local laws might be expensive.

Such tradeoffs can affect the entire software lifecycle, from conceptualization and development to deployment and operation. Many Societal Computing issues stem from recent trends in social computing, and possibly may even be solved by drawing on social computing models, such as the wisdom of crowds, collaborative filtering and so on; but many of the concerns are orthogonal and could possibly be addressed by novel approaches grounded in the SE/PL communities. We feel that the SE/PL community has a special role to play as it provides the substrate - the languages, compilers, design techniques, architectures, testing approaches, etc. on which all software systems are founded.

We describe our motivation in the next section and briefly outline some initial Societal Computing topics in the following sections. We then highlight a few research challenges posed by tensions between prospective technologies targeting these subareas.

6.1 Motivation

Anthony Kronman in his book “Education’s End: Why Our Colleges and Universities Have Given Up on the Meaning of Life” opines that graduate programs at universities have become increasingly specialized. He argues that initially universities were much more broader in their scope and increasing emphasis on the research ideal has resulted in them becoming very specialized. He says: “Graduate students learn to restrict their attention to a single segment of human knowledge and to accept their incompetence to assess, or even understand, the work of specialists in other areas. [. . .] They are taught to understand that only by accepting the limits of specialization can they ever hope to make an “original contribution” to the ever-growing body of scholarship in which the fruits on research are contained.” [86]

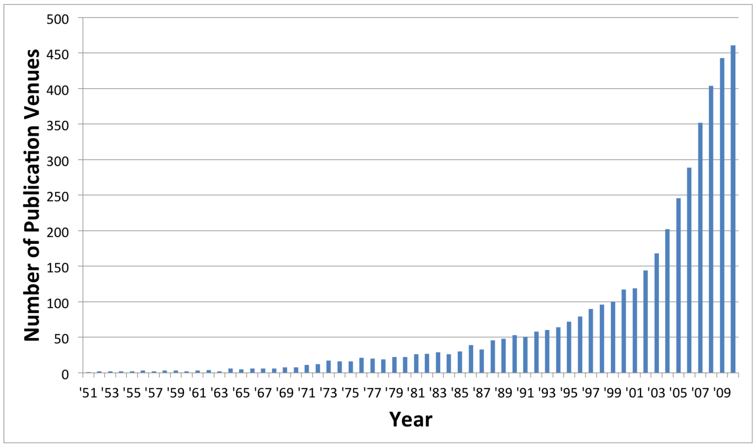


Figure 6.1: Number of Publication Venues in the ACM Digital Library from 1951 to 2010

In the field of Computer Science as well, this increasing specialization is evident by the increasing number of publication venues that exist now and by how this number has changed over the years. A good indicator of the number of publication venues is the number of proceedings (for conferences and workshops) that are available in the ACM Digital Library [2]. This is shown in Figure 6.1. We see an exponential increase in the number of the publication venues in the last ten years. As the number of publication venues has increased, it has resulted in specialization of Computer Science into subareas and sub-subareas.

While this research specialization is important and has resulted in our increased understanding of the field, it has also made our scopes very narrow. Our problem is an inverse to that of being a "jack of all trades and a master of none." Researchers have become experts in their specialized subareas (and sub-subareas) on Computer Science while being relatively unaware of the other subareas. Due to this narrowing of scope, researchers are not very aware of the advances made in the other subareas and in particular, the tradeoffs that might exist between them. Advanced research and progress made in one research subarea may have a negative effect on some other research subarea. This notion of tradeoffs is analogous to the concept of Pareto Efficiency [116] in Economics, which deals with the distribution of goods among a set of individuals in society. Pareto Efficiency refers to the state of distribution where it's not possible to make an individual "better off" without making some other individual "worse off." Not being in a Pareto Efficient state would imply that it is possible to optimize both (or multiple) areas; being in a Pareto Efficient state would imply that it is not possible to optimize one area without affecting the other one. In our case of Societal Computing, identifying such a state will be an important research challenge and this identification may not be possible without a detailed understanding of the different areas that we're trying to optimize.

We feel that such tradeoffs exist in many different areas and that a broadening of research scope is necessary to effectively address them. We need to take a much more holistic view of research. We describe some subareas of Societal Computing and the tradeoffs among them in the following sections.

6.2 Societal Computing Topics

In this section, we describe some research areas relevant to Societal Computing and we will highlight the tradeoffs among these areas in Section 6.3.

6.2.1 Privacy

Privacy in the context of social computing systems has become a major concern for the society at large. A search for the pair of terms "facebook" and "privacy" gives nearly two billion hits on popular search engines. Recent feature enhancements and policy changes in social networking and recommender applications – as well as their increasingly common use – have exacerbated this issue [25, 56, 78, 176].

With many online systems that range from providing purchasing recommendations to suggesting plausible friends, as well as media attention (e.g., the AOL anonymity-breaking incident reported by the New York Times [14]), both users and non-users of the systems (e.g., friends, family, co-workers, etc. mentioned or photographed by users) are growing more and more concerned about their personal privacy [135].

Social computing systems, when treated in combination, have created a threat that we call "Correlation Privacy." Narayanan and Shmatikov [107] demonstrated a relatively straightforward method to breach privacy and identify individuals by correlating anonymized Netflix movie rating data with public IMDb data. A similar approach could potentially be applied to any combination of such data-gathering systems, so how to safeguard again these "attacks" may be a fruitful research direction. This is analogous to earlier work addressing queries on census data but, at that time, there were relatively few prospective attackers [4, 17]. There has been some initial work towards retaining privacy while still benefiting from recommendation systems (e.g., [20, 136]). There have also been other approaches such as k-anonymity [144], differential privacy [43], and applications of differential privacy to different domains [124, 129].

A related challenge is to make the existence of privacy threats more understandable to ordinary users who do not have a technical background and/or in cases where it's not very clear how users' information might be employed by the system, particularly germane for systems that provide APIs making it easier (than screen scraping) for third parties to utilize that information (e.g., [11, 50, 87]). There has been some recent work (e.g., [79, 139]) towards this end. One interesting option might be to make privacy more quantifiable, perhaps by introducing a notion of "Gullibility Factor" for privacy settings, say ranging from 0 to 1 with 1 being the least private. For example, we could say that the default settings for Facebook have a Gullibility Factor of 0.5, whereas for Twitter it's 0.2 (we made up these numbers). A simple scoring scheme might steer away fearful users, while encouraging the merely puzzled to consult one of the numerous "how to" guide articles on privacy settings from sources such as the New York Times [126] and the BBC [125].

While research efforts in this area have been promising, there is a lot of scope for further research. Researchers will need to be aware, in particular, of the tradeoff of Privacy with other Societal Computing topics and we elaborate on this in Section 6.3.

6.2.2 Cultural Differences

Different regions and cultures around the world vary in their notions and perceptions on what is acceptable and what shouldn't be allowed. An increasingly important area of Societal Computing will be building systems that can adapt automatically to different regions and cultures.

One important cultural difference we highlight here is Legal Challenges. Legal issues present additional research challenges for a wide range of software systems beyond just social computing applications. For example, privacy-related laws vary tremendously from country to country, e.g., consider Germany compared to the United States. Systems such as Facebook and Google Street View, which have been accepted by the government and individuals in the US, are facing many obstacles in Germany [135]. After the Second World War, Germany has legislated very strict privacy laws to prevent the government from persecuting its citizens. It is illegal in Germany to publish names or images of private individuals (including felons) without their permission [110]. Before allowing Google's Street View service, German data protection agencies asked Google to audit the information collected by their street view mapping cars. During the audit, they discovered that the cars were collecting personal information such as emails and phone numbers from unsecured wifi networks [15].

One intriguing example open research problem would consider the implications of regulation diversity on software. To deal with different cultures and customs, we would require novel modularization mechanisms beyond those employed for software localizations of keyboard, written language, the customs of different geographical regions, etc. We call this "Regulatory Localization." We feel that multidisciplinary research with other areas of Computer Science such as natural language processing would be highly beneficial.

Another example of a cultural difference is censorship. Different regions believe different things should be censored. For certain topics there is almost universal agreement on censorship (e.g., child pornography [156]); some others are more debatable (e.g., political or government secrets [134], web search results [22, 157], marijuana [66]). What's acceptable would depend a lot on the region and cultural preconceptions in place. As we build software systems that have users all over the world, we would need novel techniques for dealing with such issues. This also relates to the current debate on Net Neutrality [167]. If, for example, we know that certain states/regions are not being net neutral, would we design, develop, test, or operate our software systems differently?

A recent paper [60] also discusses how local laws can affect software engineering. Those authors focus on intellectual property laws and licensing, warranty and liability, and transborder data flows, and propose “lawful software engineering” research directions concerned with coping with the wide variety of legal constraints during software development and deployment. While that work falls within the scope of our proposal, we are primarily concerned with the potential interactions with other aspects of Societal Computing.

6.2.3 Green Computing

Green Computing (or Green IT) is “the study and practice of designing, manufacturing, using, and disposing of computers, servers, and associated subsystems [...] efficiently and effectively with minimal or no impact on the environment” [106]. With our oil reserves projected to exhaust in less than fifty years [160], and renewable energy sources still providing only a small fraction [47], Green Computing here and now is becoming more and more important and, indeed, vital to our children and grandchildren.

Investigating how to build greener software systems from an SE/PL perspective, in addition to the complementary algorithmic efficiency and systems perspective such as resource allocation, platform virtualization, and power management pursued by other computer science subdisciplines [100] will be important. Aspects of green computing that are concerned about hardware, recycling, and so on are beyond the scope of this thesis; we focus specifically on software related challenges. For instance, say we could quantify complex software systems’ behavior in terms of energy expended. There has been some initial research in this area such as [165], which tries to quantify the carbon footprint of a Google Search. Then we could investigate ways to make this quantification more modular, devise software architectures and design patterns intended to give developers and end-users more control over energy use, and invent testing methods that check for energy violations. And, further, rethink testing in general, perhaps pushing more testing into the field (“perpetual testing” [113]), to reduce pre-deployment energy consumption and, perhaps, better spread the burden across energy sources. If we could make this quantification more modular (perhaps to the level of individual functionality provided by large systems), we could then provide easy means for operators and end-users to disable unneeded modules, which may play a critical role in Green Computing, to reduce energy consumption on server farms, desktops, and the increasingly abundant mobile platforms. As a simple example, a system like Netflix could inform each user that it would save X amount of energy to disable automated recommendations and only enable them when and if really needed (note that user altruism is a very different kind of model than charging extra for certain functionality [111]). But quantifying which user-visible functionality saves how much energy may not be easy, particularly when systems are built by integrating components.

Our Societal Computing initiative envisions investigating the tradeoffs of Green Computing with the other areas and we highlight these tradeoffs in the following section.

6.3 Societal Computing Tradeoffs

While there is a lot of potential for novel research in these individual areas of Societal Computing, in this thesis we focus on the tradeoffs between these different areas and the research challenges that arise out of these tensions. A central discussion point is to consider the problem of how software methodologies and technologies aimed at reducing societal costs in one area can sometimes raise societal costs for another. For example, there may be clever ways to engineer social computing and other applications to protect privacy or enforce regulations that inherently consume vast CPU cycles and other resources, which could be considered "anti-green." We need a holistic view.

6.3.1 Privacy vs. Green Computing

Say we have developed an awesome new social computing system S whose privacy-preservation properties may be suspect. One possible approach would be to try correlating S with other popular social systems, such as Netflix, IMDb, Facebook, Amazon, etc., to determine whether privacy can indeed be breached and to what degree (e.g., are potentially all users at risk, only those who use a specific other social system, or only a small fraction of the latter with unique information). We might do this prior to public use of S, e.g., using an internal test team and/or informed beta testers (who might invent phony identities). Such an experiment could give us an estimate of the likely privacy breaches, and possibly point towards steps that could be taken to safeguard against them.

However, straightforward mechanisms that poke or data-mine for potential breaches would likely require substantial computational resources; while this kind of testing may be a good idea where Correlation Privacy is concerned, it may not be so good for Green Computing. And it also does not address correlation against future social computing systems or unexpected uses of our system. So instead we could wait until S has been populated by the general public and then periodically correlate a sampling, which might require fewer resources and/or better distribute the resource burden, as well as draw on other new social systems as they are launched. But by then any privacy threats could be actual rather than hypothetical, and consequent protective measures too late. What design and testing techniques can we devise to balance privacy with green computing, particularly in a context where subsystems might be developed by different organizations? Broadening of research scope will be important to be able to effectively address these concerns.

6.3.2 Privacy vs. Cultural Differences

As countries are increasingly trying to pass new privacy laws [16, 99] and companies are being taken to court and getting fined for privacy violations [36, 102], legal issues dealing with privacy will become even more complex. We believe that as countries mandate new requirements for privacy, there will be an important tradeoff between these laws and privacy issues. Say we have an awesome new system S that has users in different parts of the world. As each country might have (slightly) different privacy laws, our system would need to comply with all the different regulations. Imagine a user Fred who is a US citizen. We would need to comply with US regulations in this case and Fred would have set his privacy settings as needed. Now if Fred decides to travel to another country (say, Germany) for business or a conference, we might also need to comply with the German regulations for privacy. In addition, we might also need to comply with the EU regulations, which may or may not be the same as the German regulations. Having to comply with all these different regulations will only end up making privacy threats and settings harder to understand for users and might also result in less usable systems. Note that such conflicts and confusions needn't arise due to travel to different countries, but might also exist due to the different city, state, and federal rules. What techniques can we use to make privacy and privacy settings more understandable to ordinary users when we need to also comply with complex legal regulations? An understanding of the different research areas involved will be crucial to address the various research challenges that we face.

6.3.3 Green Computing vs. Green Computing

There is also an interesting (and recursive) tradeoff of Green Computing with itself. As part of the development of greener software systems, we may need to invest substantial computer resources. For example, social recommendation systems tend to rely on expensive data-mining, but developing a greener recommendation system that is kinder to the environment could also be quite expensive. In the worst case, the amount of resources spent on building such green systems may far outweigh the energy benefits of replacing their less-green counterparts with these new systems, a classic example of being "penny wise, pound foolish." How can we efficiently analyze this in advance of expending those resources?

6.4 Related Work

Social Computing has become increasingly popular in recent times. It aims to leverage community knowledge and participation to improve individual interactions. The rise of Web 2.0 and social networking has popularized social computing as a research area. Applications of social computing have ranged from recommender systems in various domains [61, 105, 131, 170] to programming languages for social computing and crowd sourcing platforms [6]. A recent article by Scekic, Truong, and Dustdar [132] tries to analyze the behavior and economic incentives of crowds in large social systems. All these papers focus on leveraging the crowd to help with a specific goal. With societal computing, on the other hand, our goal is to help address the tradeoffs and longer-term problems for society at large.

In the field of software engineering, social software engineering has gained increasing traction. Social software engineering focuses on leveraging the "socialness of software" [93] to improve the software engineering process. There has been a lot of work that integrates social aspects into the daily software engineering lifecycle [19,70,130,158]. There have been many studies that have tried to aid software developers by increasing awareness of what others in the team are doing [57,81,89,149]. These papers have largely focused on improving the software engineering experience for the software developers. On the other hand, our target audience is the entire society at large and not specifically software developers.

A recent paper by German, Webber, and Di Penta [60] focuses on IP laws and how laws affect software engineering. While this falls within the scope of societal computing, our focus is on the tradeoffs and interactions with other domains.

6.5 How can other research contribute?

A common theme in these tradeoffs is finding the right balance between the different areas of Societal Computing. If we haven't reached the Pareto Efficient state yet, it might be possible to optimize different areas simultaneously. Once we reach the Pareto Efficient state, trying to improve one of these areas might have an adverse effect of some other area. An important concern and a big research challenge will be trying to identify such a state - would this be pair-wise for the different subareas? would this be multi-variable across all possible areas? We believe that this will require a detailed understanding of the various Societal Computing areas. What to do once we reach the Pareto Efficient state gives us further food for thought. One approach to consider, even though it might be considered an anathema to all technological advances, is to spend more *human* time to reduce reliance on non-renewable resources. Most technology (since the dawn of time) has been designed to make humans more productive and to reduce the burden of work for humans. However, as resources start becoming scarce, humans may need to take on more of this burden. This might imply a greater reliance on design or code review instead of execution testing. We would then need to figure out how we could do reviews across different systems, e.g., to manually find Correlation Privacy problems. We might also encourage human policing of privacy violations and/or time spent in end-user training to reduce the Gullibility Factor rather than automated ways for detecting these.

One argument towards Societal Computing might be that many different communities - such as operating systems and networks - need to look at these problems as well. We agree that multi-disciplinary research is crucial and we feel that the SE/PL community needs to expand its scope towards more multidisciplinary research efforts. As the rest of the CS community usually ends up writing software to implement their research ideas and put them into practice, we have the special (and perhaps enviable?) role of cutting across the various domains as we provide the underlying platform — i.e., languages, compilers, development techniques, etc. — for implementation for most systems. Naturally it behooves us to come up with ways of dealing with Societal Computing concerns.

We could make available means such as design patterns, architectural metaphors, better tools, APIs, smarter compilers, better testing techniques, and new programming languages to deal with some of these concerns. We can help the other communities make an easier decision when it comes to the tradeoffs. We can also address how to implement these balanced systems. We feel that a broadening of research scope is very important and necessary to address the research challenges and in particular, the tradeoffs among the different areas of Societal Computing. Finding the right balance among the tradeoffs in these different research areas will be crucial.

6.6 How can this thesis contribute?

As far as this thesis is concerned, there are many parts of it that can aid in addressing some of these tradeoffs. In Chapter 2, we describe a large online survey of privacy requirements. The survey instrument, which is provided in Appendix A, can be reused and extended for future studies. Our results can also serve as a benchmark for comparing the data. This can help build a body of knowledge and provide guidelines such as best practices. Further, depending on the domain, our survey and experimental design can serve as a blueprint and template for designing newer surveys and studies. These studies could aim to gather requirements from users on what their concerns are for topic X and the best techniques for mitigating these concerns.

In Chapters 3 and 4, we used crowdsourcing to design new tools that would make privacy better for end-users from two points of view: designing friendlier, easy to understand tools for privacy settings and using software testing for detecting privacy bugs. Similar systems can be designed that use crowdsourcing to address other user concerns for topics of Societal Computing. In general, designing user friendly systems will be key to make users better aware of the tradeoffs. To verify how good these newer systems are, our study design (described in Appendix B) can serve as a template. Whenever software systems are built for addressing topics of Societal Computing, these systems would need to be tested. In addition to the traditional software testing techniques, our crowdsourced technique could also be used. This could be done by end-users after the software is deployed, thus being similar to perpetual testing [113].

In Chapter 5, we described an approach for getting privacy without any additional computational overhead. Our approach used the accidental side-effect of doing some already existing computation in many systems. We would not have discovered this had it not been for multidisciplinary research that we were conducting at that time. Thus, we feel that collaborations with people in other domains will become increasingly crucial for making these kinds of discoveries. From a more technical point of view, more emphasis should be placed on Mechanism Discovery (in addition to the already existing Mechanism Design). A lot of focus is spent on coming up with new algorithms or solutions for a certain problem; perhaps in many cases, existing algorithms in other domains might help address some open problems and help with some of the tradeoffs. Our approach of combining two known techniques can serve as a template for future tradeoffs in Societal Computing.

Chapter 7

Conclusions

Privacy in computing systems has become very important in recent years. However, there is very little evidence and empirical results about what exactly are the user concerns and how to mitigate them. In particular, in the software engineering community, there have been no systematic studies for privacy requirements and there has been very little work on software testing for privacy.

In this thesis, the first project focused on gathering privacy requirements using a large online survey. The goal was to explore privacy requirements for users and developers in online systems such as Amazon and Facebook. Our study gave us key insights into the differences between users and software developers and between people from North America, Europe, and Asia. These insights and guidelines can help create a framework on privacy requirements for software developers when designing and building such software systems.

One of the key insights from the user study showed that, as far as users are concerned, more transparency and providing details on how data is used is as effective as more "technical" solutions (such as anonymization) for mitigating privacy concerns. The second part of the thesis used this insight to create and evaluate a new technique for improving users' understanding of how their data is shared on social networks like Facebook. Our approach used crowdsourcing to provide users a means of comparison of what they share versus other subsets of users such as their friends or colleagues. A majority of the participants, after trying out our crowdsourcing tool for privacy, said that they would prefer such a tool over the current settings available in Facebook.

An added benefit of our crowdsourcing tool was that it can help detect privacy bugs from a software testing perspective. Our tool detects inconsistencies between what is viewable on the website versus via the API, and system-wide policy changes on making certain information public or private. For the third part of the thesis, we demonstrated how and what kinds of bugs it can detect and showed some real-world examples of bugs found in Facebook.

Recent work on privacy shows that trying to address privacy concerns typically results in a large system overhead in terms of CPU computation time. For the fourth part of the thesis, we describe a technique to mitigate privacy concerns without incurring any CPU overhead. We were able to leverage the similarity between two algorithms to provide privacy as an accidental side-effect of doing some already existing computation. Thus, we did not need to spend any additional CPU resources for privacy.

Finally, we generalized this notion of tradeoffs as many such tradeoffs occur when building large complex software systems. We call this "Societal Computing", a new research area that is concerned with the impact of computational tradeoffs on societal issues such as privacy, green computing, climate change, cultural differences, and sustainability. We described how the results and artifacts from all our projects on privacy can be applied and reused for exploring the tradeoffs in other areas of Societal Computing.

Bibliography

[1] Oxford English Dictionary Online. `http://www.oed.com/`, September 2013.

[2] ACM. ACM Proceedings. `http://dl.acm.org/proceedings.cfm`, 2011.

[3] Alessandro Acquisti, Leslie John, and George Loewenstein. What is privacy worth? In *Workshop on Information Systems and Economics (WISE)*, 2009.

[4] Nabil R. Adam and John C. Worthmann. Security-control methods for statistical databases: a comparative study. *ACM Comput. Surv.*, 21(4):515–556, 1989.

[5] Dakshi Agrawal and Charu C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. In *PODS '01: Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 247–255, New York, NY, USA, 2001. ACM.

[6] Salman Ahmad, Alexis Battle, Zahan Malkani, and Sepander Kamvar. The jabberwocky programming environment for structured social computing. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pages 53–64, New York, NY, USA, 2011. ACM.

[7] Alex Koppel. Koala. `https://github.com/arsduo/koala/`, April 2010.

[8] Terry Anderson and Heather Kanuka. E-research: Methods, strategies, and issues. 2003.

[9] Julia Angwin and Jennifer Valentino-Devries. FTC Backs Do-Not-Track System for Web. `http://online.wsj.com/article/SB10001424052748704594804575648670826747094.html`, December 2010.

[10] Apache. Mahout. `http://mahout.apache.org/`.

[11] Apple Developer. iOS Dev Center. `http://developer.apple.com/devcenter/ios/index.action`, 2007.

[12] Lars Backstrom, Cynthia Dwork, and Jon Kleinberg. Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 181–190, New York, NY, USA, 2007. ACM.

[13] Liana B. Baker and Jim Finkle. Sony PlayStation suffers massive data breach. `http://www.reuters.com/article/2011/04/26/us-sony-stoldendata-idUSTRE73P6WB20110426`, April 2011.

[14] M. Barbaro, T. Zeller, and S. Hansell. A face is exposed for AOL searcher no. 4417749. *New York Times*, 9, 2006.

[15] BBC. German Street View goes live with enhanced privacy. `http://www.bbc.co.uk/news/technology-11673117`, November 2010.

[16] BBC. Governments 'not ready' for new European privacy law. `http://www.bbc.co.uk/news/technology-12677534`, March 2011.

[17] Leland L. Beck. A security mechanism for statistical database. *ACM Trans. Database Syst.*, 5(3):316–3338, 1980.

[18] Justin Lee Becker and Hao Chen. *Measuring privacy risk in online social networks.* PhD thesis, University of California, Davis, 2009.

[19] Andrew Begel, Khoo Yit Phang, and Thomas Zimmermann. Codebook: discovering and exploiting relationships in software repositories. In *ICSE '10: Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering*, pages 125–134, New York, NY, USA, 2010. ACM.

[20] S. Berkovsky, Y. Eytani, T. Kuflik, and F. Ricci. Enhancing privacy and preserving accuracy of a distributed collaborative filtering. In *RecSys '07: Proc. of the 2007 ACM conf. on Recommender systems*, pages 9–16, 2007.

[21] Smriti Bhagat, Graham Cormode, Balachander Krishnamurthy, and Divesh Srivastava. Privacy in dynamic social networks. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 1059–1060, New York, NY, USA, 2010. ACM.

[22] Bloomberg News. Baidu Sued in New York Court for Censoring China Internet Search Results. `http://www.bloomberg.com/news/2011-05-18/baidu-com-accused-in-u-s-lawsuit-of-aiding-chinese-internet-censorship.html`, May 2011.

[23] Avrim Blum, Cynthia Dwork, Frank McSherry, and Kobbi Nissim. Practical privacy: the sulq framework. In *PODS '05: Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 128–138, New York, NY, USA, 2005. ACM.

[24] Avrim Blum, Katrina Ligett, and Aaron Roth. A learning theory approach to non-interactive database privacy. In *STOC '08: Proceedings of the 40th annual ACM symposium on Theory of computing*, pages 609–618, New York, NY, USA, 2008. ACM.

[25] Bianca Bosker. Facebook CEO 'Doesn't Believe In Privacy'. `http://www.huffingtonpost.com/2010/04/29/zuckerberg-privacy-stance_n_556679.html`, April 2010.

[26] Brad Fitzpatrick. LiveJournal. `http://www.livejournal.com/`, 1999.

[27] Travis D. Breaux and Annie I. Anton. Analyzing regulatory rules for privacy and security requirements. *IEEE Transactions on Software Engineering*, 34(1):5–20, 2008.

[28] Travis D. Breaux and Ashwini Rao. Formal analysis of privacy requirements specifications for multi-tier applications. In *RE'13: Proceedings of the 21st IEEE International Requirements Engineering Conference (RE'13)*, Washington, DC, USA, July 2013. IEEE Society Press.

[29] Manfred Broy. The 'grand challenge' in informatics: Engineering software-intensive systems. *Computer*, 39(10):72–80, 2006.

[30] Raymond P. L. Buse and Thomas Zimmermann. Information Needs for Software Development Analytics. In *Proceedings of the 2012 International Conference on Software Engineering*, ICSE 2012, pages 987–996, Piscataway, NJ, USA, 2012. IEEE Press.

[31] Gerardo Canfora, Luigi Cerulo, Marta Cimitile, and Massimiliano Di Penta. Social interactions around cross-system bug fixings: the case of freebsd and openbsd. In *Proceeding of the 8th working conference on Mining software repositories*, MSR '11, pages 143–152, New York, NY, USA, 2011. ACM.

[32] Miguel Castro, Manuel Costa, and Jean-Philippe Martin. Better bug reporting with better privacy. In *Proceedings of the 13th international conference on Architectural support for programming languages and operating systems*, ASPLOS XIII, pages 319–328, New York, NY, USA, 2008. ACM.

[33] James Clause and Alessandro Orso. Camouflage: automated anonymization of field data. In *Proceeding of the 33rd international conference on Software engineering*, ICSE '11, pages 21–30, New York, NY, USA, 2011. ACM.

[34] E. Cohen and M. Strauss. Maintaining time-decaying stream aggregates. In *Proc. of the 22nd ACM SIGMOD-SIGACT-SIGART symposium on principles of database systems (PODS)*, pages 223–233, 2003.

[35] Jon Cohen. Most Americans back NSA tracking phone records, prioritize probes over privacy. `http://www.washingtonpost.com/politics/most-americans-support-nsa-tracking-phone-records-prioritize-investigations-over-privacy/2013/06/10/51e721d6-d204-11e2-9f1a-1a7cdee20287_story.html`, June 2013.

[36] Noam Cohen. It's Tracking Your Every Move and You May Not Even Know. `http://www.nytimes.com/2011/03/26/business/media/26privacy.html`, March 2011.

[37] Lorrie Faith Cranor and Norman Sadeh. A shortage of privacy engineers. *Security & Privacy, IEEE*, 11(2):77–79, 2013.

[38] T. Dalenius. Towards a methodology for statistical disclosure control. *Statistik Tidskrift*, 15:429–444, 1977.

[39] Yi Ding and Xue Li. Time weight collaborative filtering. In *CIKM '05: Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 485–492, New York, NY, USA, 2005. ACM.

[40] Irit Dinur and Kobbi Nissim. Revealing information while preserving privacy. In *PODS '03: Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 202–210, New York, NY, USA, 2003. ACM.

[41] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. *Theory of Cryptography*, pages 265–284, 2006.

[42] C. Dwork and K. Nissim. Privacy-preserving datamining on vertically partitioned databases. *Lecture Notes in Computer Science*, pages 528–544, 2004.

[43] Cynthia Dwork. Differential privacy. *IN ICALP*, 2:1–12, 2006.

[44] Cynthia Dwork, Moni Naor, Omer Reingold, Guy N. Rothblum, and Salil Vadhan. On the complexity of differentially private data release: efficient algorithms and hardness results. In *STOC '09: Proceedings of the 41st annual ACM symposium on Theory of computing*, pages 381–390, New York, NY, USA, 2009. ACM.

[45] Michael D. Ekstrand, Michael Ludwig, Joseph A. Konstan, and John T. Riedl. Rethinking the recommender research ecosystem: reproducibility, openness, and lenskit. In *Proceedings of the fifth ACM conference on Recommender systems*, RecSys '11, pages 133–140, New York, NY, USA, 2011. ACM.

[46] William Enck, Peter Gilbert, Byung-Gon Chun, Landon P. Cox, Jaeyeon Jung, Patrick McDaniel, and Anmol N. Sheth. Taintdroid: an information-flow tracking system for realtime privacy monitoring on smartphones. In *Proc. of the 9th USENIX Conf. on Operating systems design and impl.*, pages 1–6, 2010.

[47] U.S. Energy Information Administration. International Energy Outlook 2010 - Highlights. `http://www.eia.doe.gov/oiaf/ieo/highlights.html`, May 2010.

[48] Steven Erlanger. Outrage in Europe Grows Over Spying Disclosures. `http://www.nytimes.com/2013/07/02/world/europe/france-and-germany-piqued-over-spying-scandal.html`, July 2013.

[49] Alexandre Evfimievski, Johannes Gehrke, and Ramakrishnan Srikant. Limiting privacy breaches in privacy preserving data mining. In *PODS '03: Proceedings of the twenty-second ACM*

SIGMOD-SIGACT-SIGART symposium on Principles of database systems, pages 211–222, New York, NY, USA, 2003. ACM.

[50] Facebook. Facebook Developers. `http://developers.facebook.com/`, 2007.

[51] Facebook. Graph API. `https://developers.facebook.com/docs/reference/api/`, 2007.

[52] Facebook. User. `https://developers.facebook.com/docs/reference/api/user/`, 2007.

[53] Facebook Developers. Bugs – Can no longer access FriendList members on test users. `https://developers.facebook.com/bugs/368623589859564`, June 2012.

[54] Facebook Developers. Bugs – Some of the friendlists do not show members from Graph API, why?? `https://developers.facebook.com/bugs/400876833291706`, April 2012.

[55] Lujun Fang and Kristen LeFevre. Privacy wizards for social networking sites. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 351–360, New York, NY, USA, 2010. ACM.

[56] Dan Fletcher. How Facebook Is Redefining Privacy. `http://www.time.com/time/business/article/0,8599,1990582.html`, May 2010.

[57] Thomas Fritz and Gail C. Murphy. Using information fragments to answer the questions developers ask. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 1*, ICSE '10, pages 175–184, New York, NY, USA, 2010. ACM.

[58] Zeno Gantner, Lucas Drumond, Christoph Freudenthaler, Steffen Rendle, and Lars Schmidt-Thieme. Learning attribute-to-feature mappings for cold-start recommendations. In *Proceedings of the 2010 IEEE International Conference on Data Mining*, ICDM '10, pages 176–185, Washington, DC, USA, 2010. IEEE Computer Society.

[59] Guilbert Gates. Facebook privacy: A bewildering tangle of options. `http://www.nytimes.com/interactive/2010/05/12/business/facebook-privacy.html`, May 2010.

[60] Daniel M. German, Jens H. Webber, and Massimiliano Di Penta. Lawful software engineering. In *Proceedings of the FSE/SDP workshop on Future of software engineering research*, FoSER '10, pages 129–132, New York, NY, USA, 2010. ACM.

[61] W. Geyer, C. Dugan, D. R. Millen, M. Muller, and J. Freyne. Recommending topics for self-descriptions in online user profiles. In *RecSys '08: Proc. of the 2008 ACM conference on Recommender systems*, pages 59–66, 2008.

[62] Vindu Goel. Facebook to update privacy policy, but adjusting settings is no easier. *The New York Times*, August 2013. Accessed October 01, 2013.

[63] Google Developers. Google+ API – Google+ Platform. `https://developers.google.com/+/api/`, 2011.

[64] Google Inc. A better web. better for the environment. `http://www.google.com/green/bigpicture/references.html`, 2012. Accessed September 30, 2013.

[65] Mark Grechanik, Christoph Csallner, Chen Fu, and Qing Xie. Is data privacy always good for software testing? *Software Reliability Engineering, International Symposium on*, 0:368–377, 2010.

[66] Ryan Grim. Facebook Blocks Ads For Pot Legalization Campaign. `http://www.huffingtonpost.com/2010/08/24/facebook-blocks-ads-for-p_n_692295.html`, August 2010.

[67] Sam Grobart. The Facebook Scare That Wasn't. `http://gadgetwise.blogs.nytimes.com/2011/08/10/the-facebook-scare-that-wasnt/`, August 2011.

[68] GroupLens Research Group. Movielens data sets. `http://grouplens.org/datasets/movielens/`.

[69] Asela Gunawardana and Christopher Meek. A unified approach to building hybrid recommender systems. In *Proceedings of the third ACM conference on Recommender systems*, RecSys '09, pages 117–124, New York, NY, USA, 2009. ACM.

[70] I. Hadar, S. Sherman, and O. Hazzan. Learning Human Aspects of Collaborative Software Development. *Journal of Information Systems Education*, 19(3):311–319, 2008.

[71] E. Hammer-Lahav, D. Recordon, and D. Hardt. The OAuth 2.0 authorization protocol. `http://tools.ietf.org/html/draft-ietf-oauth-v2/`, 2010.

[72] Mary Jean Harrold, James A. Jones, Tongyu Li, Donglin Liang, Alessandro Orso, Maikel Pennings, Saurabh Sinha, S. Alexander Spoon, and Ashish Gujarathi. Regression test selection for java software. In *Proc. of the 16th ACM SIGPLAN Conf. on OO prog., systems, languages, and appl.*, pages 312–326, 2001.

[73] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22:5–53, January 2004.

[74] Francis Heylighen and Johan Bollen. Hebbian algorithms for a digital library recommendation system. *Parallel Processing Workshops, International Conference on*, 0:439, 2002.

[75] Jacob Jacoby and Michael S. Matell. Three-point likert scales are good enough. *Journal of Marketing Research*, 8(4):pp. 495–500, 1971.

[76] Vilas Jagannath, Qingzhou Luo, and Darko Marinov. Change-aware preemption prioritization. In *Proceedings of the 2011 International Symposium on Software Testing and Analysis*, ISSTA '11, pages 133–143, 2011.

[77] M. Johnson, S. Egelman, and S.M. Bellovin. Facebook and privacy: It's complicated. In *Symp. on Usable Privacy and Security*, 2012.

[78] Steven Johnson. Web Privacy: In Praise of Oversharing. `http://www.time.com/time/business/article/0,8599,1990586.html`, May 2010.

[79] Patrick Gage Kelley, Lucian Cesca, Joanna Bresee, and Lorrie Faith Cranor. Standardizing privacy notices: an online study of the nutrition label approach. In *Proceedings of the 28th international conference on Human factors in computing systems*, CHI '10, pages 1573–1582, New York, NY, USA, 2010. ACM.

[80] John Paul Kelly and Derek Bridge. Enhancing the diversity of conversational collaborative recommendations: a comparison. *Artif. Intell. Rev.*, 25:79–95, April 2006.

[81] Mik Kersten and Gail C. Murphy. Using task context to improve programmer productivity. In *Proceedings of the 14th ACM SIGSOFT international symposium on Foundations of software engineering*, SIGSOFT '06/FSE-14, pages 1–11, New York, NY, USA, 2006. ACM.

[82] Ralf Klinkenberg. Learning drifting concepts: Example selection vs. example weighting. *Intell. Data Anal.*, 8(3):281–300, 2004.

[83] Andrew J. Ko, Robin Abraham, Laura Beckwith, Alan Blackwell, Margaret Burnett, Martin Erwig, Chris Scaffidi, Joseph Lawrance, Henry Lieberman, Brad Myers, Mary Beth Rosson, Gregg Rothermel, Mary Shaw, and Susan Wiedenbeck. The state of the art in end-user software engineering. *ACM Comput. Surv.*, 43(3):21:1–21:44, April 2011.

[84] Yehuda Koren. Collaborative filtering with temporal dynamics. *Commun. ACM*, 53(4):89–97, 2010.

[85] I. Koychev and I. Schwab. Adaptation to drifting user's interests. In *Proceedings of ECML2000 Workshop: Machine Learning in New Information Age*, pages 39–46. Citeseer, 2000.

[86] A.T. Kronman. *Education's End: Why Our Colleges and Universities Have Given Up on the Meaning of Life.* Yale University Press, 2007.

[87] Last.fm. API. `http://www.last.fm/api`, 2009.

[88] Neal Lathia, Stephen Hailes, and Licia Capra. Private distributed collaborative filtering using estimated concordance measures. In *RecSys '07: Proceedings of the 2007 ACM conference on Recommender systems*, pages 1–8, New York, NY, USA, 2007. ACM.

[89] Soo Ling Lim, Daniele Quercia, and Anthony Finkelstein. Stakenet: using social networks to analyse the stakeholders of large-scale software projects. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 1*, ICSE '10, pages 295–304, New York, NY, USA, 2010. ACM.

[90] Heather Richter Lipford, Andrew Besmer, and Jason Watson. Understanding privacy settings in facebook with an audience view. *UPSEC*, 8:1–8, 2008.

[91] Yabing Liu, Krishna P. Gummadi, Balachander Krishnamurthy, and Alan Mislove. Analyzing facebook privacy settings: user expectations vs. reality. In *Proc. of the 2011 SIGCOMM Conf. on Internet measurement conf.*, pages 61–70, 2011.

[92] W. Maalej, T. Fritz, and R. Robbes. Collecting and processing interaction data for recommendation systems. In M. Robillard, M. Maalej, R. Walker, and T. Zimmerman, editors, *Recommendation Systems in Software Engineering*, pages 173–197. Springer, 2014.

[93] Walid Maalej and Dennis Pagano. On the socialness of software. In *Proceedings of the International Software on Social Computing and its Applications*. IEEE Computer Society, 2011.

[94] Michelle Madejski, Maritza Johnson, and Steven M. Bellovin. A study of privacy settings errors in an online social network. *Pervasive Computing and Comm. Workshops, IEEE Intl. Conf. on*, 0:340–345, 2012.

[95] Clara Mancini, Yvonne Rogers, Keerthi Thomas, Adam N. Joinson, Blaine A. Price, Arosha K. Bandara, Lukasz Jedrzejczyk, and Bashar Nuseibeh. In the best families: Tracking and relationships. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 2419–2428, New York, NY, USA, 2011. ACM.

[96] D. Mashima, E. Shi, R. Chow, P. Sarkar, C. Li, and D. Song. Privacy settings from contextual attributes: A case study using google buzz. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2011 IEEE International Conference on*, pages 257–262, 2011.

[97] Frank McSherry and Ilya Mironov. Differentially private recommender systems: building privacy into the net. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, pages 627–636, New York, NY, USA, 2009. ACM.

[98] Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *FOCS '07: Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, pages 94–103, Washington, DC, USA, 2007. IEEE Computer Society.

[99] Joseph Menn. White House calls for online privacy law. `http://www.ft.com/cms/s/0/7267c2c4-500d-11e0-9ad1-00144feab49a.html`, March 2011.

[100] Microsoft. *Green Computing*, volume 18. The Architecture Journal, 2008.

[101] Claire Cain Miller. Privacy Officials Worldwide Press Google About Glass. http://bits.blogs.nytimes.com/2013/06/19/privacy-officials-worldwide-press-google-about-glass/, June 2013.

[102] Claire Cain Miller and Tanzina Vega. Google Introduces New Social Tool and Settles Privacy Charge. http://www.nytimes.com/2011/03/31/technology/31ftc.html, March 2011.

[103] Ilya Mironov, Omkant Pandey, Omer Reingold, and Salil Vadhan. Computational differential privacy. In *Advances in Cryptology-CRYPTO 2009*, pages 126–142. Springer, 2009.

[104] H. Muccini, A. Di Francesco, and P. Esposito. Software testing of mobile applications: Challenges and future research directions. In *Automation of Software Test (AST), 2012 7th International Workshop on*, pages 29–35, June 2012.

[105] C. Murphy, S. Sheth, G. Kaiser, and L. Wilcox. genSpace: Exploring Social Networking Metaphors for Knowledge Sharing and Scientific Collaborative Work. In *1st Intl. Workshop on Social Software Engg. and Applications*, pages 29–36, September 2008.

[106] S. Murugesan. Harnessing green it: Principles and practices. *IT Professional*, 10(1):24 –33, jan.-feb. 2008.

[107] Arvind Narayanan and Vitaly Shmatikov. How to break anonymity of the netflix prize dataset. *CoRR*, abs/cs/0610105, 2006.

[108] National Academy of Engineering. Grand challenges. http://www.engineeringchallenges.org/cms/challenges.aspx.

[109] John Nunemaker, Wynn Netherland, Erik Michaels-Ober, and Steve Richert. The Twitter Ruby Gem. http://twitter.rubyforge.org/, 2006.

[110] Kevin O'Brien. Technology Butts Up Against Germany's Privacy Laws. http://www.nytimes.com/2010/07/12/technology/12disconnect.html, July 2010.

[111] Andrew Odlyzko. A modest proposal for preventing internet congestion. Technical report, AT&T Labs - Research, 1997.

[112] Inah Omoronyia, Luca Cavallaro, Mazeiar Salehie, Liliana Pasquale, and Bashar Nuseibeh. Engineering adaptive privacy: On the role of privacy awareness requirements. In *Proceedings of the 2013 International Conference on Software Engineering*, ICSE '13, pages 632–641, Piscataway, NJ, USA, 2013. IEEE Press.

[113] L. Osterweil. Perpetually testing software. In *Proc. of the The Ninth International Software Quality Week*, May 1996.

[114] Jeremiah Owyang. Coping With Twitter's Unfollow Bug. `http://techcrunch.com/2012/03/27/unfollowbug/`, March 2012.

[115] Janak J. Parekh, Ke Wang, and Salvatore J. Stolfo. Privacy-preserving payload-based correlation for accurate malicious traffic detection. In *Proc. of the SIGCOMM Workshop on Large-scale attack defense*, pages 99–106, 2006.

[116] Vilfredo Pareto. The new theories of economics. *The Journal of Political Economy*, 5(4):pp. 485–502, 1897.

[117] Seung-Taek Park and Wei Chu. Pairwise preference regression for cold-start recommendation. In *Proceedings of the third ACM conference on Recommender systems*, RecSys '09, pages 21–28, New York, NY, USA, 2009. ACM.

[118] Thomas Paul, Martin Stopczynski, Daniel Puscher, Melanie Volkamer, and Thorsten Strufe. C4ps: colors for privacy settings. In *Proceedings of the 21st international conference companion on World Wide Web*, WWW '12 Companion, pages 585–586, New York, NY, USA, 2012. ACM.

[119] Sarah Perez. Facebook Wins "Worst API" in Developer Survey. `http://techcrunch.com/2011/08/11/facebook-wins-worst-api-in-developer-survey/`, August 2011.

[120] F. Peters and T. Menzies. Privacy and utility for defect prediction: Experiments with morph. In *Software Engineering (ICSE), 2012 34th International Conference on*, pages 189–199. IEEE, 2012.

[121] H. Polat and Wenliang Du. Privacy-preserving collaborative filtering using randomized perturbation techniques. In *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*, pages 625–628, Nov. 2003.

[122] Nathaniel Popper and Somini Sengupta. U.S. Says Ring Stole 160 Million Credit Card Numbers. `http://dealbook.nytimes.com/2013/07/25/arrests-planned-in-hacking-of-financial-companies/`, July 2013.

[123] Xiao Qu, Myra B. Cohen, and Gregg Rothermel. Configuration-aware regression testing: an empirical study of sampling and prioritization. In *Proc. of the 2008 Intl. Symp. on Software testing and analysis*, pages 75–86, 2008.

[124] Jason Reed, Adam J. Aviv, Daniel Wagner, Andreas Haeberlen, Benjamin C. Pierce, and Jonathan M. Smith. Differential privacy for collaborative security. In *Proceedings of the Third European Workshop on System Security*, EUROSEC '10, pages 1–7, New York, NY, USA, 2010. ACM.

[125] LJ Rich. A guide to protecting your privacy on Facebook. `http://news.bbc.co.uk/2/hi/programmes/click_online/8717750.stm`, June 2010.

[126] Riva Richmond. Gadgetwise: A Guide to Facebook's New Privacy Settings. `http://gadgetwise.blogs.nytimes.com/2010/05/27/5-steps-to-reset-your-facebook-privacy-settings/`, May 2010.

[127] Ralph L Rosnow and Robert Rosenthal. *Beginning behavioral research: A conceptual primer .* Prentice-Hall, Inc, 1996.

[128] Aaron Roth and Tim Roughgarden. Interactive privacy via the median mechanism. In *STOC '10: Proceedings of the 42nd ACM symposium on Theory of computing*, pages 765–774, New York, NY, USA, 2010. ACM.

[129] Indrajit Roy, Srinath T. V. Setty, Ann Kilzer, Vitaly Shmatikov, and Emmett Witchel. Airavat: security and privacy for MapReduce. In *Proceedings of the 7th USENIX conference on Networked systems design and implementation*, NSDI'10, pages 20–20, Berkeley, CA, USA, 2010. USENIX Association.

[130] Anita Sarma, Larry Maccherone, Patrick Wagstrom, and James Herbsleb. Tesseract: Interactive visual exploration of socio-technical relationships in software development. In *Proceedings of*

the 31st International Conference on Software Engineering, ICSE '09, pages 23–33, Washington, DC, USA, 2009. IEEE Computer Society.

[131] Zachary M. Saul, Vladimir Filkov, Premkumar Devanbu, and Christian Bird. Recommending random walks. In *Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering*, ESEC-FSE '07, pages 15–24, New York, NY, USA, 2007. ACM.

[132] Ognjen Scekic, Hong-Linh Truong, and Schahram Dustdar. Incentives and rewarding in social computing. *Commun. ACM*, 56(6):72–82, June 2013.

[133] E.J. Schwartz, T. Avgerinos, and D. Brumley. All you ever wanted to know about dynamic taint analysis and forward symbolic execution (but might have been afraid to ask). In *Security and Privacy (SP), 2010 IEEE Symposium on*, pages 317–331. IEEE, 2010.

[134] Israel Shamir. The Guardian's Political Censorship of Wikileaks. `http://www.counterpunch.org/2011/01/11/the-guardian-s-political-censorship-of-wikileaks/`, January 2011.

[135] Maggie Shiels. Germany officials launch legal action against Facebook. `http://news.bbc.co.uk/2/hi/technology/8798906.stm`, July 2010.

[136] Reza Shokri, Pedram Pedarsani, George Theodorakopoulos, and Jean-Pierre Hubaux. Preserving privacy in collaborative filtering through distributed aggregation of offline profiles. In *RecSys '09: Proceedings of the third ACM conference on Recommender systems*, pages 157–164, New York, NY, USA, 2009. ACM.

[137] V.K. Singh, H. Schulzrinne, and K. Miao. Dyswis: An architecture for automated diagnosis of networks. In *Network Operations and Management Symposium*, pages 851–854. IEEE, 2008.

[138] Daniel J Solove. A Taxonomy of Privacy. *University of Pennsylvania Law Review*, pages 477–564, 2006.

[139] S. Spiekermann and L.F. Cranor. Engineering privacy. *Software Engineering, IEEE Transactions on*, 35(1):67 –82, jan.-feb. 2009.

[140] Malte Spitz. Germans Loved Obama. Now We Don't Trust Him. `http://www.nytimes.com/2013/06/30/opinion/sunday/germans-loved-obama-now-we-dont-trust-him.html`, June 2013.

[141] Richard C Sprinthall and Stephen T Fisk. *Basic statistical analysis.* Prentice Hall Englewood Cliffs, NJ, 1990.

[142] Anna Cinzia Squicciarini, Mohamed Shehab, and Federica Paci. Collective privacy management in social networks. In *Proceedings of the 18th international conference on World wide web*, WWW '09, pages 521–530, New York, NY, USA, 2009. ACM.

[143] Fred Stutzman, Ralph Gross, and Alessandro Acquisti. Silent listeners: The evolution of privacy and disclosure on facebook. *Journal of Privacy and Confidentiality*, 4(2):2, 2013.

[144] Latanya Sweeney. k-anonymity: a model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10(5):557–570, 2002.

[145] Kunal Taneja, Mark Grechanik, Rayid Ghani, and Tao Xie. Testing software in age of data privacy: a balancing act. In *Proceedings of the 19th ACM SIGSOFT symposium and the 13th European conference on Foundations of software engineering*, SIGSOFT/FSE '11, pages 201–211, New York, NY, USA, 2011. ACM.

[146] Christine Task and Chris Clifton. A guide to differential privacy theory in social network analysis. In *Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2012)*, pages 411–417. IEEE Computer Society, 2012.

[147] Amin Tootoonchian, Stefan Saroiu, Yashar Ganjali, and Alec Wolman. Lockr: better privacy for social networks. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, CoNEXT '09, pages 169–180, New York, NY, USA, 2009. ACM.

[148] Vincent Toubiana, Vincent Verdot, Benoit Christophe, and Mathieu Boussard. Photo-tape: user privacy preferences in photo tagging. In *Proceedings of the 21st international conference companion on World Wide Web*, WWW '12 Companion, pages 617–618, New York, NY, USA, 2012. ACM.

[149] Christoph Treude and Margaret-Anne Storey. Awareness 2.0: staying aware of projects, developers and tasks using dashboards and feeds. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 1*, ICSE '10, pages 365–374, New York, NY, USA, 2010. ACM.

[150] Thein Than Tun, A.K. Bandara, B.A. Price, Yijun Yu, C. Haley, I. Omoronyia, and B. Nuseibeh. Privacy arguments: Analysing selective disclosure requirements for mobile applications. In *Requirements Engineering Conference (RE), 2012 20th IEEE International*, pages 131–140, Sept 2012.

[151] Tracy L Tuten, David J Urban, and Michael Bosnjak. Internet surveys and data quality: A review. *Online social sciences*, page 7, 2000.

[152] Twitter. Twitter Developers. `https://dev.twitter.com/`, 2008.

[153] Twitter. About Public and Protected Tweets. `http://support.twitter.com/entries/14016`, 2012.

[154] Twitter Developers. Twitter Libraries. `https://dev.twitter.com/docs/twitter-libraries/`, 2012.

[155] Twitter Developers. Users. `https://dev.twitter.com/docs/platform-objects/users`, 2012.

[156] UNICEF. Optional Protocol on the sale of children, child prostitution and child pornography. `http://www.unicef.org/crc/index_30204.html`, June 2011.

[157] Jessica Vascellaro and Loretta Chao. Google Defies China on Web. `http://online.wsj.com/article/SB10001424052748704117304575137960803993890.html`, March 2010.

[158] Gina Venolia, John Tang, Ruy Cervantes, Sara Bly, George Robertson, Bongshin Lee, and Kori Inkpen. Embodied social proxy: mediating interpersonal connection in hub-and-satellite teams. In *CHI '10: Proceedings of the 28th international conference on Human factors in computing systems*, pages 1049–1058, New York, NY, USA, 2010. ACM.

[159] Vassilios S. Verykios, Elisa Bertino, Igor Nai Fovino, Loredana Parasiliti Provenza, Yucel Saygin, and Yannis Theodoridis. State-of-the-art in privacy preserving data mining. *SIGMOD Rec.*, 33(1):50–57, 2004.

[160] John Vidal. The end of oil is closer than you think. `http://www.guardian.co.uk/science/2005/apr/21/oilandpetrol.news`, April 2005.

[161] Samuel D Warren and Louis D Brandeis. The Right to Privacy. *Harvard law review*, pages 193–220, 1890.

[162] Alan F. Westin. Science, privacy, and freedom: Issues and proposals for the 1970's. part i–the current impact of surveillance on privacy. *Columbia Law Review*, 66(6):pp. 1003–1050, 1966.

[163] Jim Whitehead. Collaboration in software engineering: A roadmap. In *2007 Future of Software Engineering*, FOSE '07, pages 214–225, Washington, DC, USA, 2007. IEEE Computer Society.

[164] G. Widmer and M. Kubat. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23(1):69–101, 1996.

[165] Alex Wissner-Gross. How you can help reduce the footprint of the Web. `http://www.timesonline.co.uk/tol/news/environment/article5488934.ece`, January 2009.

[166] G. Wondracek, T. Holz, E. Kirda, and C. Kruegel. A practical attack to de-anonymize social network users. In *Security and Privacy (SP), 2010 IEEE Symposium on*, pages 223–238, 2010.

[167] Edward Wyatt. Court Rejects Suit on Net Neutrality Rules. `http://www.nytimes.com/2011/04/05/technology/05net.html`, April 2011.

[168] S. Yoo and M. Harman. Regression testing minimization, selection and prioritization: a survey. *Softw. Test. Verif. Reliab.*, 22(2):67–120, March 2012.

[169] Alyson L. Young and Anabel Quan-Haase. Information revelation and internet privacy concerns on social network sites: a case study of facebook. In *Proceedings of the fourth international conference on Communities and technologies*, C&#38;T '09, pages 265–274, New York, NY, USA, 2009. ACM.

[170] V. Zanardi and L. Capra. Social ranking: uncovering relevant content using tag-based recommender systems. In *RecSys '08: Proc. of the 2008 ACM Conf. on Recommender systems*, pages 51–58, 2008.

[171] J. Zhang and P. Pu. A recursive prediction algorithm for collaborative filtering recommender systems. In *RecSys '07: Proc. of the 2007 ACM conference on Recommender systems*, pages 57–64, 2007.

[172] Lu Zhang, Shan-Shan Hou, Chao Guo, Tao Xie, and Hong Mei. Time-aware test-case prioritization using integer linear programming. In *Proc. of the 2009 Intl. Symp. on Software testing and analysis*, pages 213–224, 2009.

[173] Mi Zhang and Neil Hurley. Statistical modeling of diversity in top-n recommender systems. In *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology - Volume 01*, WI-IAT '09, pages 490–497, Washington, DC, USA, 2009. IEEE Computer Society.

[174] Elena Zheleva and Lise Getoor. To join or not to join: the illusion of privacy in social networks with mixed public and private user profiles. In *Proceedings of the 18th international conference on World wide web*, WWW '09, pages 531–540, New York, NY, USA, 2009. ACM.

[175] Yun Zhu, Li Xiong, and Christopher Verdery. Anonymizing user profiles for personalized web search. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 1225–1226, New York, NY, USA, 2010. ACM.

[176] Mark Zuckerberg. Making Control Simple. `http://blog.facebook.com/blog.php?post=391922327130`, May 2010.

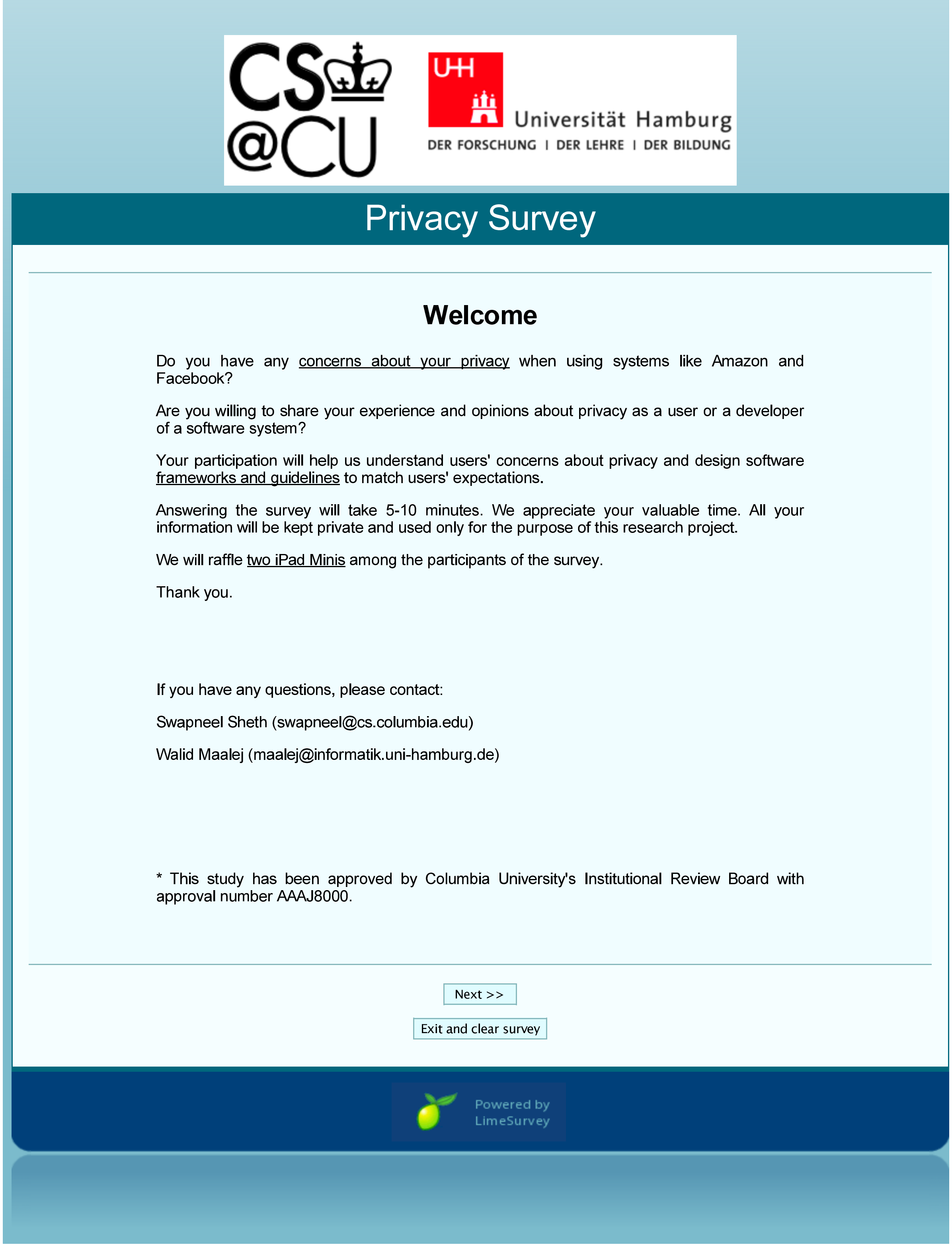
Appendices

Appendix A

Privacy Requirements User Study

The four versions of the survey used for the user study are shown next.

A.1 Version for Software Developers (English)



Privacy Survey

Welcome

Do you have any concerns about your privacy when using systems like Amazon and Facebook?

Are you willing to share your experience and opinions about privacy as a user or a developer of a software system?

Your participation will help us understand users' concerns about privacy and design software frameworks and guidelines to match users' expectations.

Answering the survey will take 5-10 minutes. We appreciate your valuable time. All your information will be kept private and used only for the purpose of this research project.

We will raffle two iPad Minis among the participants of the survey.

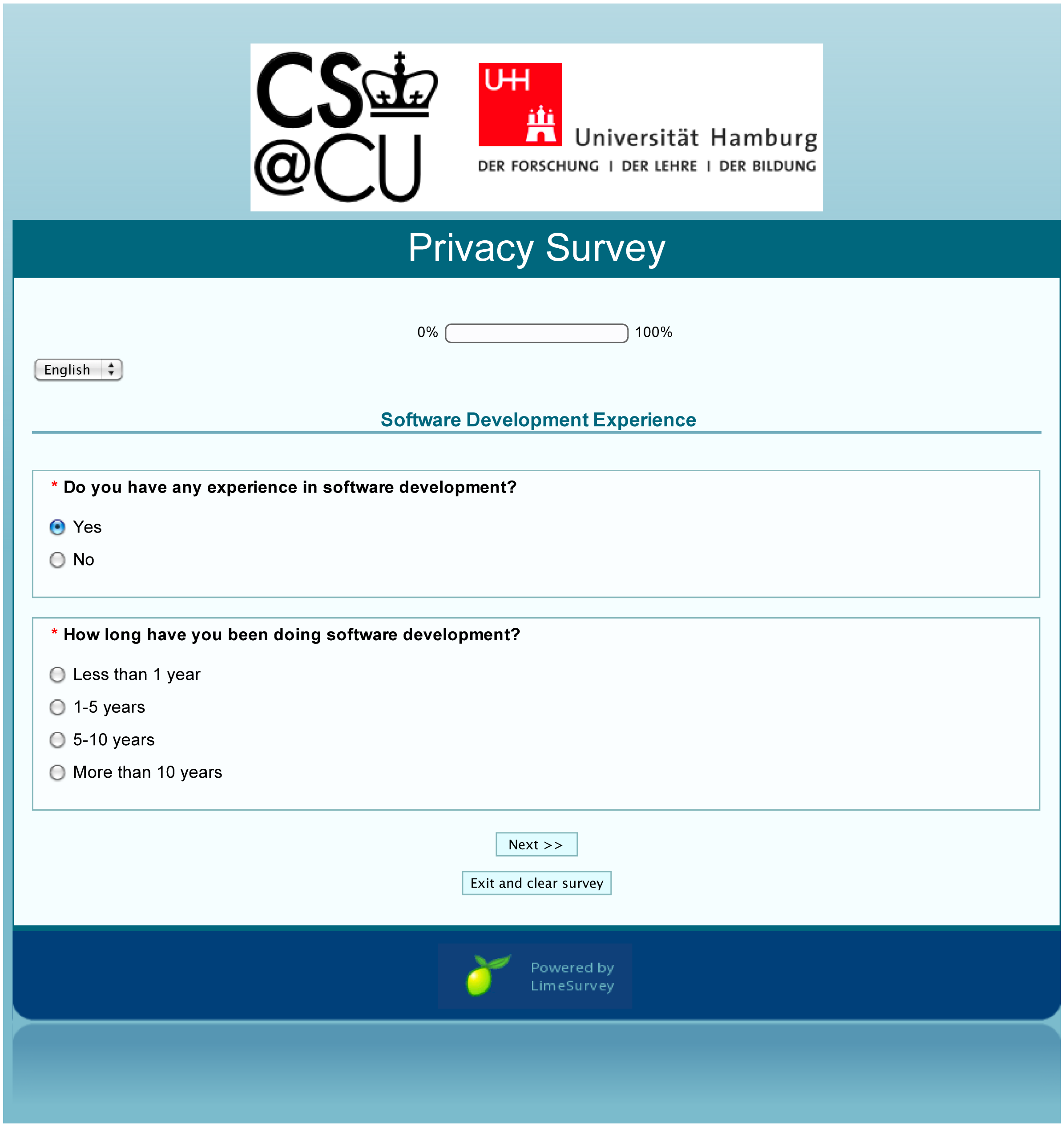
Thank you.

If you have any questions, please contact:

Swapneel Sheth (swapneel@cs.columbia.edu)

Walid Maalej (maalej@informatik.uni-hamburg.de)

* This study has been approved by Columbia University's Institutional Review Board with approval number AAAJ8000.



Privacy Survey

Software Development Experience

* Do you have any experience in software development?

● Yes

○ No

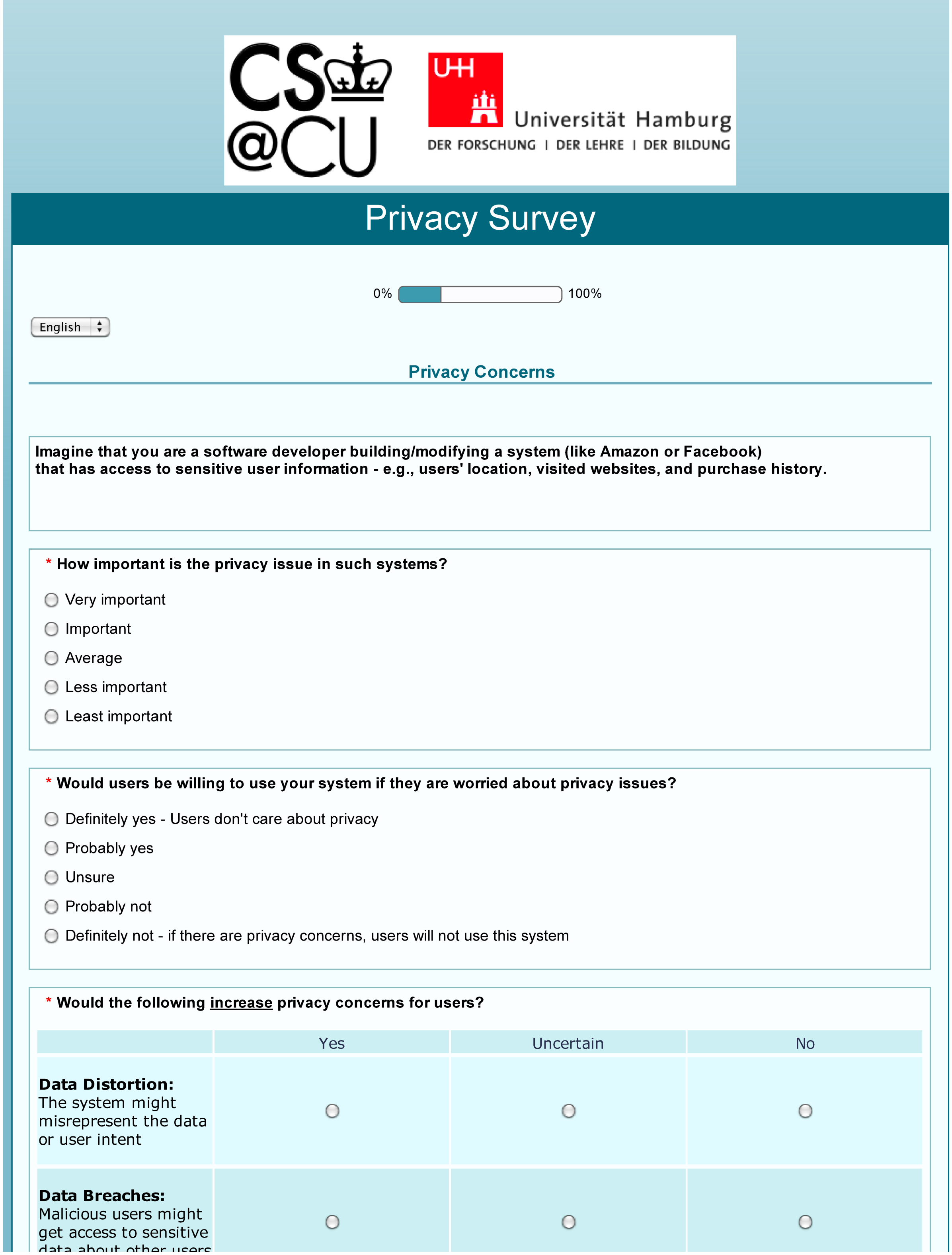
* How long have you been doing software development?

○ Less than 1 year

○ 1-5 years

○ 5-10 years

○ More than 10 years



Privacy Survey

Privacy Concerns

Imagine that you are a software developer building/modifying a system (like Amazon or Facebook) that has access to sensitive user information - e.g., users' location, visited websites, and purchase history.

* **How important is the privacy issue in such systems?**

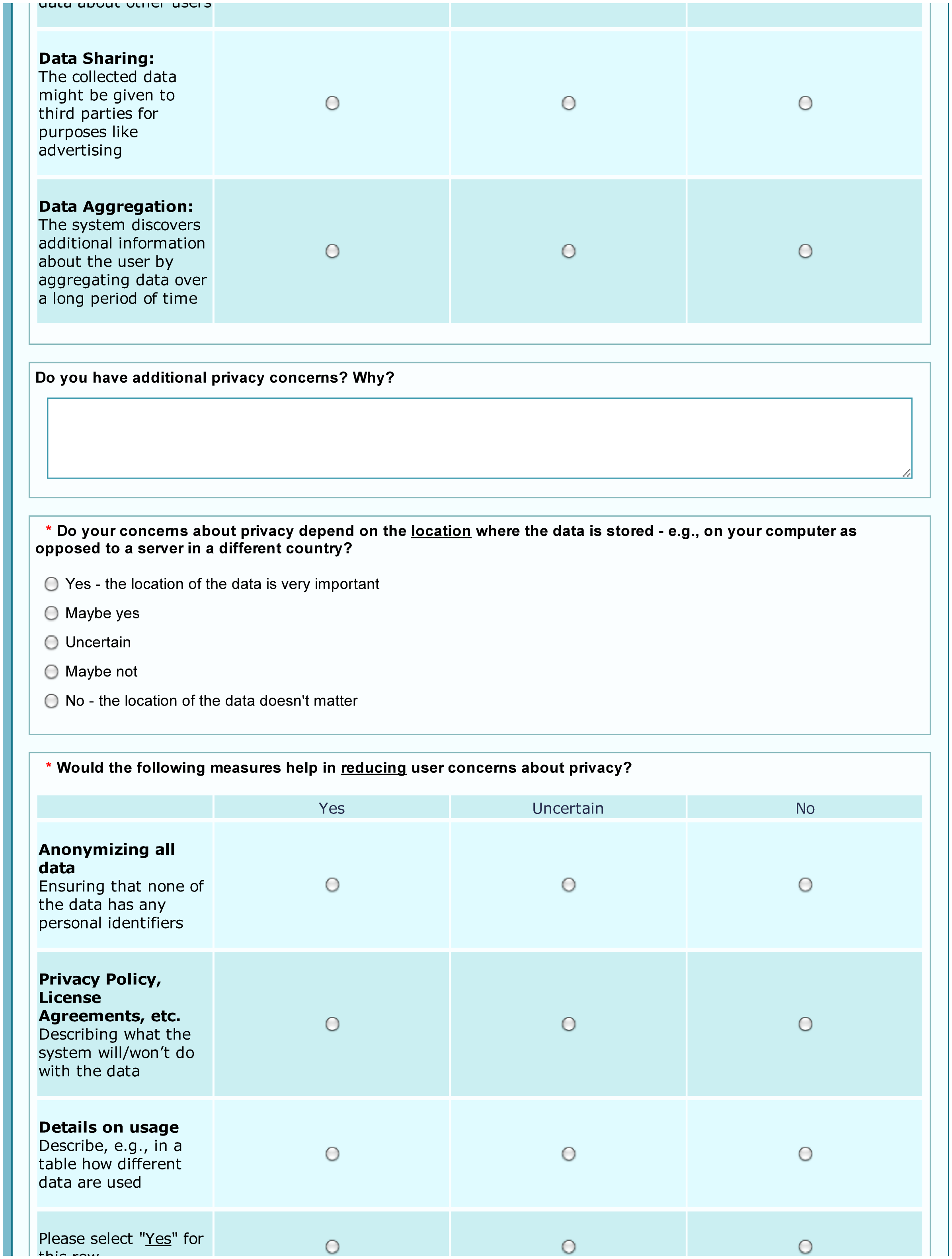
○ Very important
○ Important
○ Average
○ Less important
○ Least important

* **Would users be willing to use your system if they are worried about privacy issues?**

○ Definitely yes - Users don't care about privacy
○ Probably yes
○ Unsure
○ Probably not
○ Definitely not - if there are privacy concerns, users will not use this system

* **Would the following increase privacy concerns for users?**

	Yes	Uncertain	No
Data Distortion: The system might misrepresent the data or user intent	○	○	○
Data Breaches: Malicious users might get access to sensitive data about other users	○	○	○



data about other users			
Data Sharing: The collected data might be given to third parties for purposes like advertising	○	○	○
Data Aggregation: The system discovers additional information about the user by aggregating data over a long period of time	○	○	○

Do you have additional privacy concerns? Why?

* **Do your concerns about privacy depend on the location where the data is stored - e.g., on your computer as opposed to a server in a different country?**

○ Yes - the location of the data is very important

○ Maybe yes

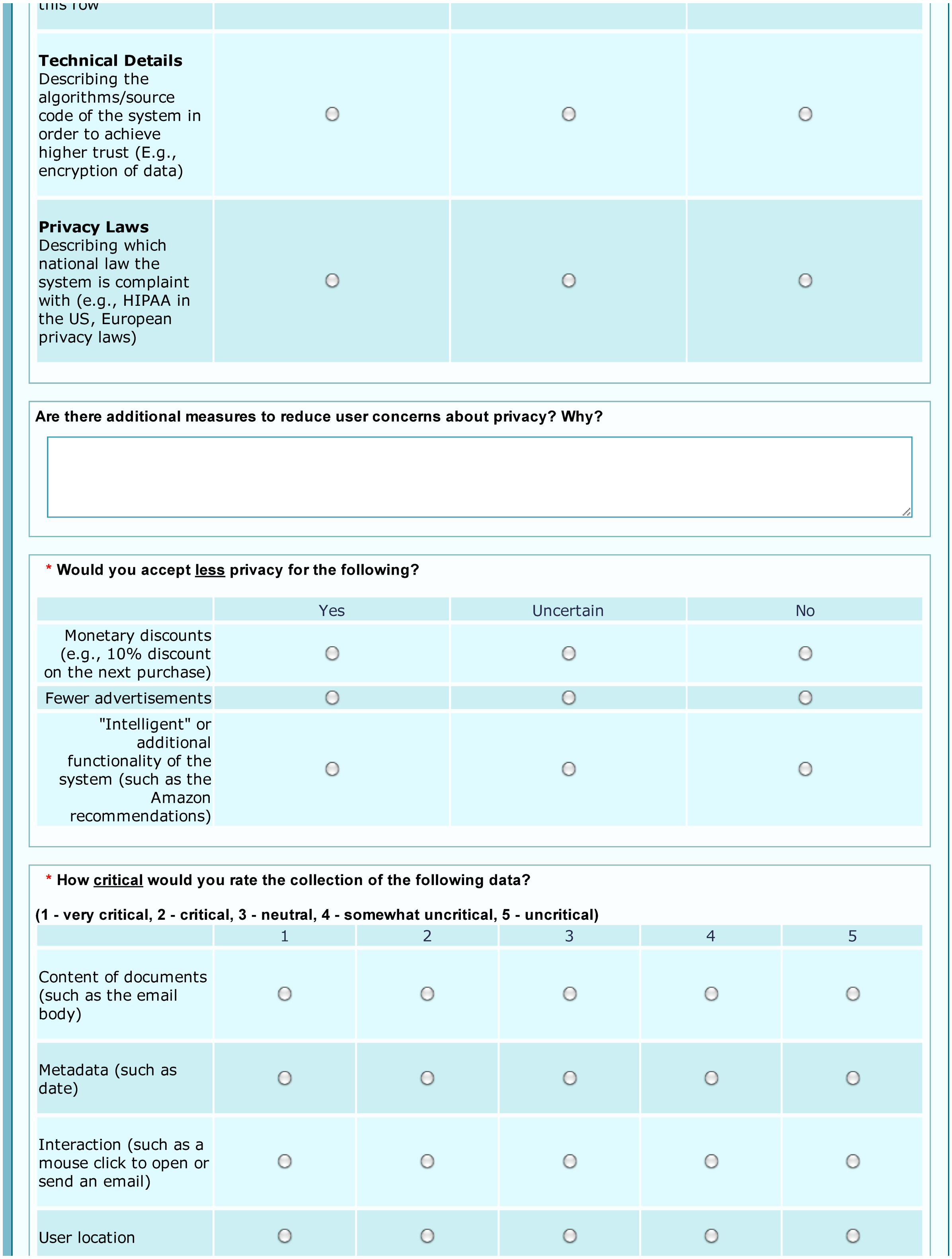
○ Uncertain

○ Maybe not

○ No - the location of the data doesn't matter

* **Would the following measures help in reducing user concerns about privacy?**

	Yes	Uncertain	No
Anonymizing all data Ensuring that none of the data has any personal identifiers	○	○	○
Privacy Policy, License Agreements, etc. Describing what the system will/won't do with the data	○	○	○
Details on usage Describe, e.g., in a table how different data are used	○	○	○
Please select "Yes" for this row	○	○	○



this row			
Technical Details Describing the algorithms/source code of the system in order to achieve higher trust (E.g., encryption of data)	○	○	○
Privacy Laws Describing which national law the system is complaint with (e.g., HIPAA in the US, European privacy laws)	○	○	○

Are there additional measures to reduce user concerns about privacy? Why?

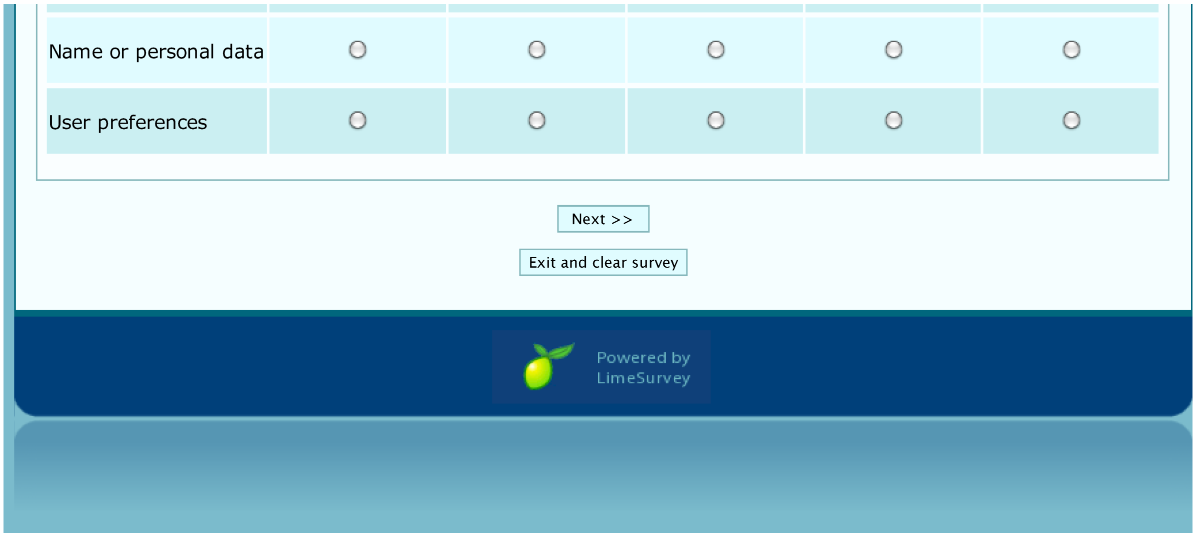
* **Would you accept less privacy for the following?**

	Yes	Uncertain	No
Monetary discounts (e.g., 10% discount on the next purchase)	○	○	○
Fewer advertisements	○	○	○
"Intelligent" or additional functionality of the system (such as the Amazon recommendations)	○	○	○

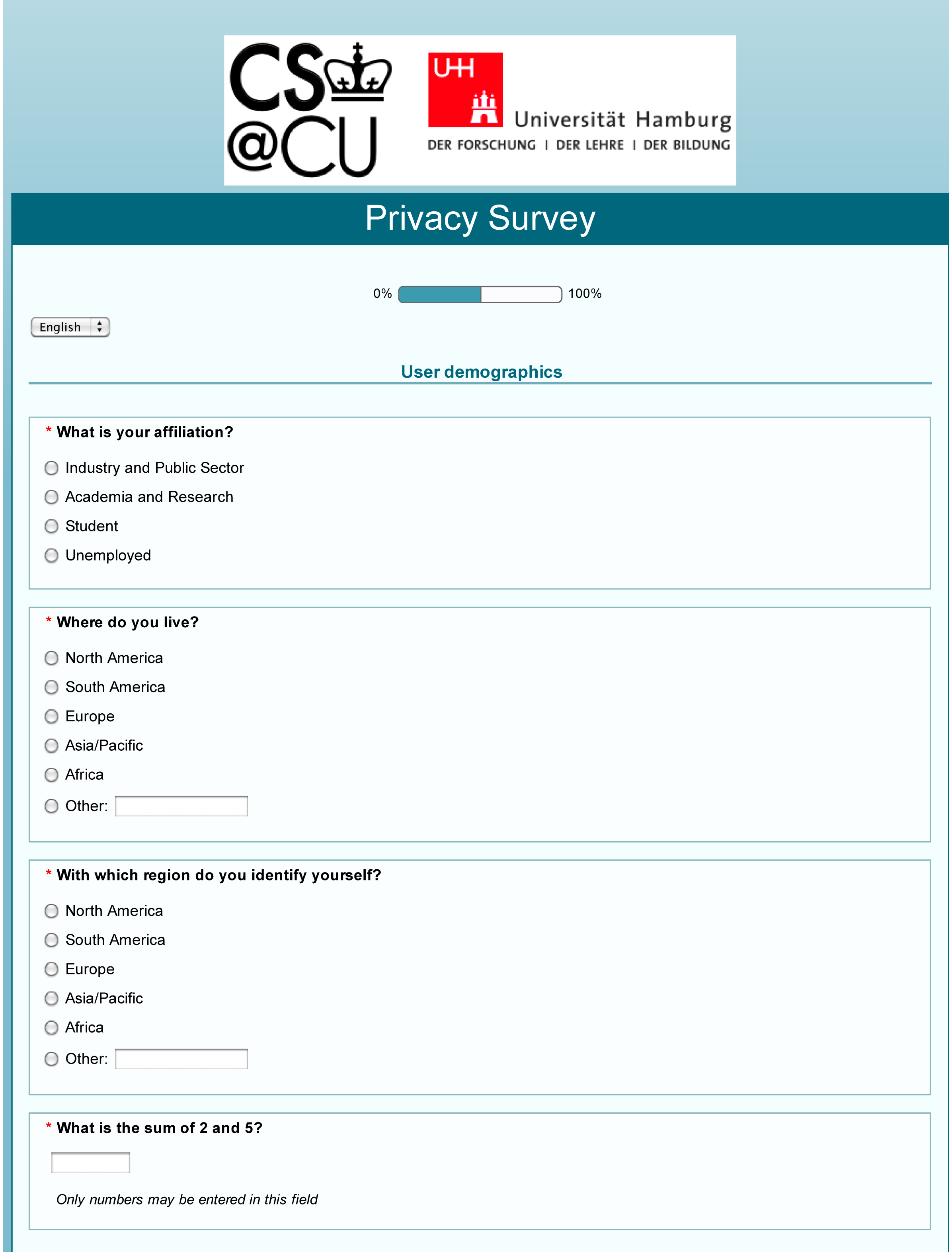
* **How critical would you rate the collection of the following data?**

(1 - very critical, 2 - critical, 3 - neutral, 4 - somewhat uncritical, 5 - uncritical)

	1	2	3	4	5
Content of documents (such as the email body)	○	○	○	○	○
Metadata (such as date)	○	○	○	○	○
Interaction (such as a mouse click to open or send an email)	○	○	○	○	○
User location	○	○	○	○	○



Name or personal data	○	○	○	○	○
User preferences	○	○	○	○	○



Privacy Survey

User demographics

* **What is your affiliation?**

○ Industry and Public Sector

○ Academia and Research

○ Student

○ Unemployed

* **Where do you live?**

○ North America

○ South America

○ Europe

○ Asia/Pacific

○ Africa

○ Other:

* **With which region do you identify yourself?**

○ North America

○ South America

○ Europe

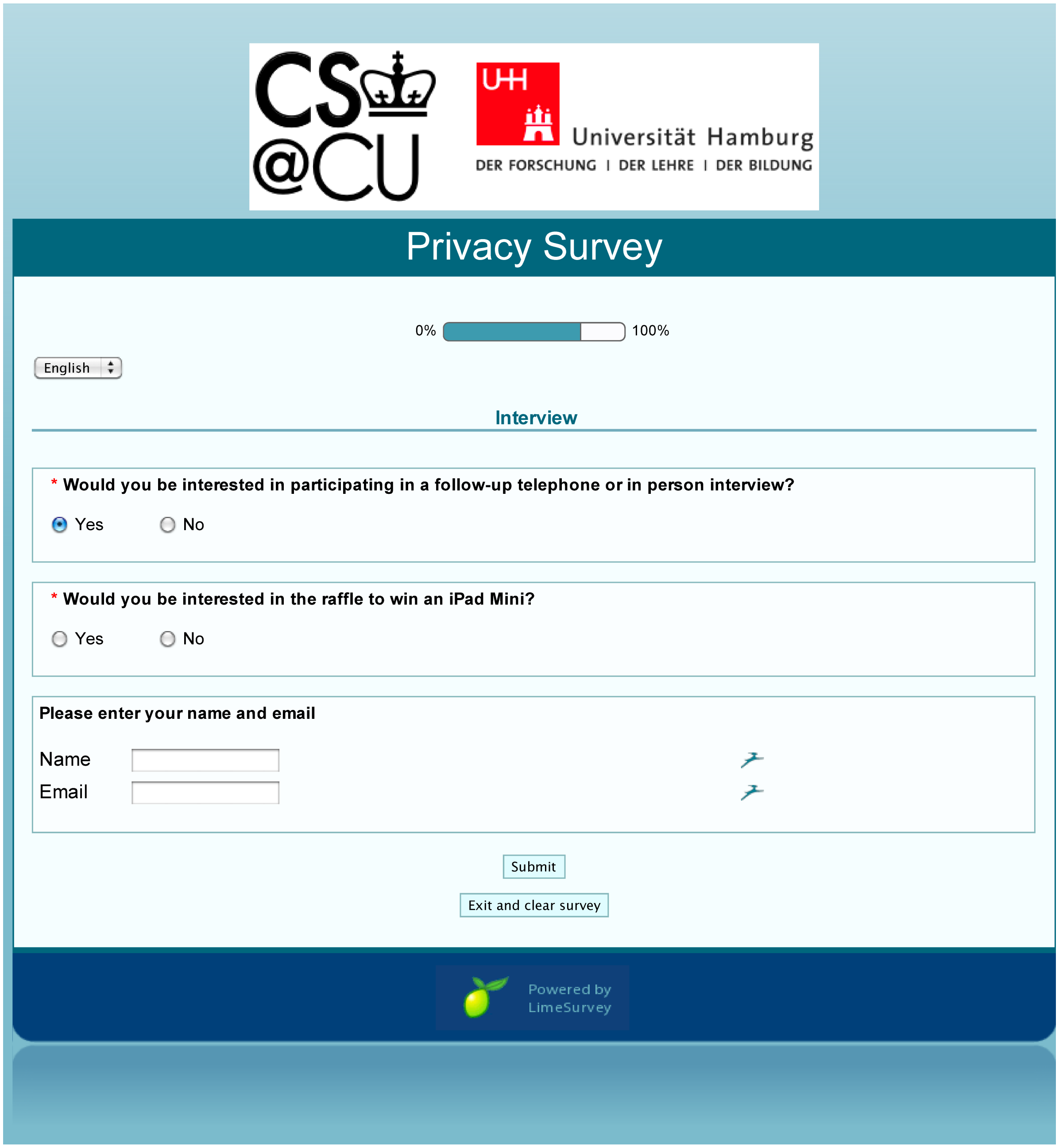
○ Asia/Pacific

○ Africa

○ Other:

* **What is the sum of 2 and 5?**

Only numbers may be entered in this field



Privacy Survey

Interview

* Would you be interested in participating in a follow-up telephone or in person interview?

● Yes ○ No

* Would you be interested in the raffle to win an iPad Mini?

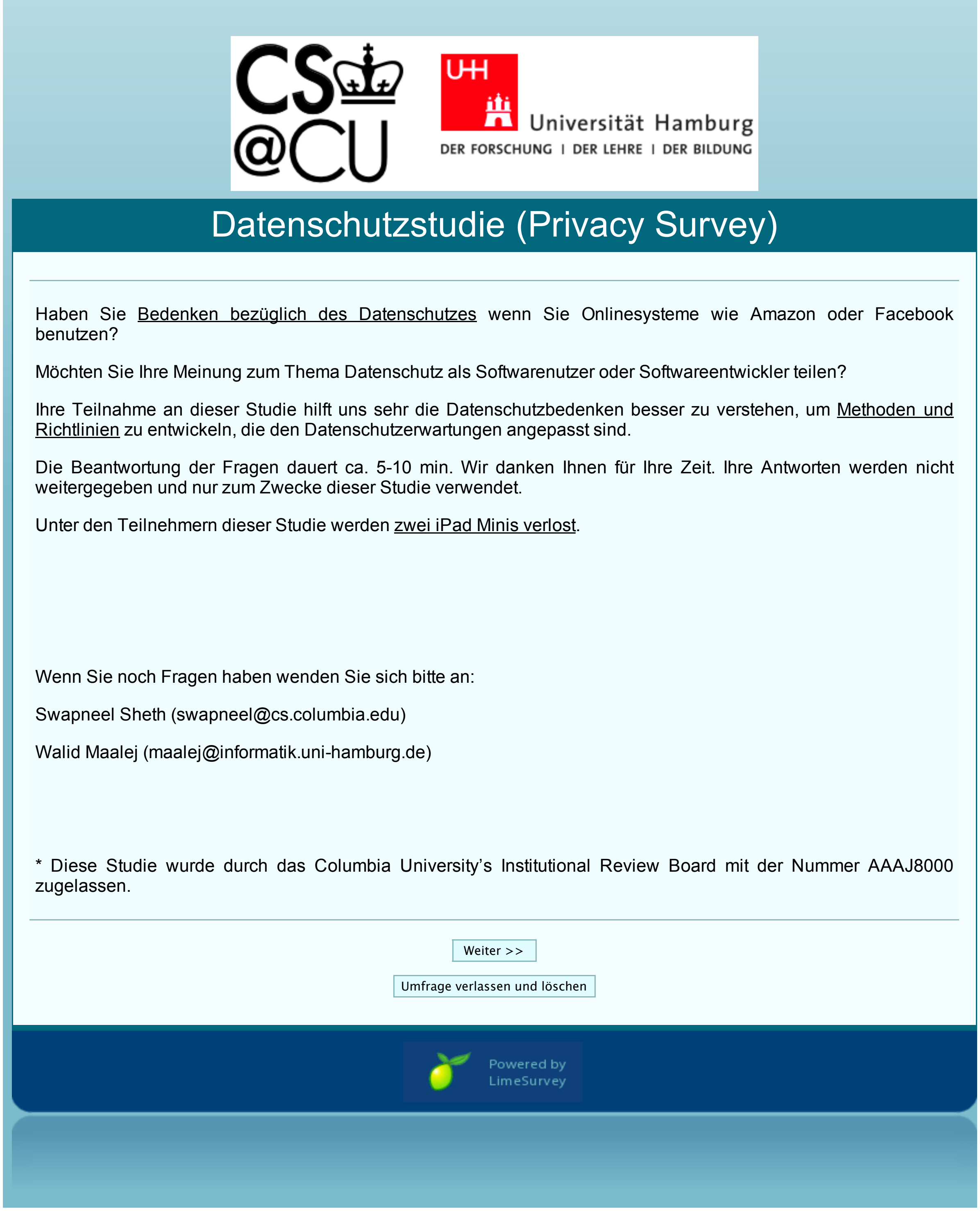
○ Yes ○ No

Please enter your name and email

Name

Email

A.2 Version for Software Developers (German)


CS @CU
UHH Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG
Weiter >>
Umfrage verlassen und löschen
Powered by LimeSurvey

Datenschutzstudie (Privacy Survey)

Haben Sie Bedenken bezüglich des Datenschutzes wenn Sie Onlinesysteme wie Amazon oder Facebook benutzen?

Möchten Sie Ihre Meinung zum Thema Datenschutz als Softwarenutzer oder Softwareentwickler teilen?

Ihre Teilnahme an dieser Studie hilft uns sehr die Datenschutzbedenken besser zu verstehen, um Methoden und Richtlinien zu entwickeln, die den Datenschutzerwartungen angepasst sind.

Die Beantwortung der Fragen dauert ca. 5-10 min. Wir danken Ihnen für Ihre Zeit. Ihre Antworten werden nicht weitergegeben und nur zum Zwecke dieser Studie verwendet.

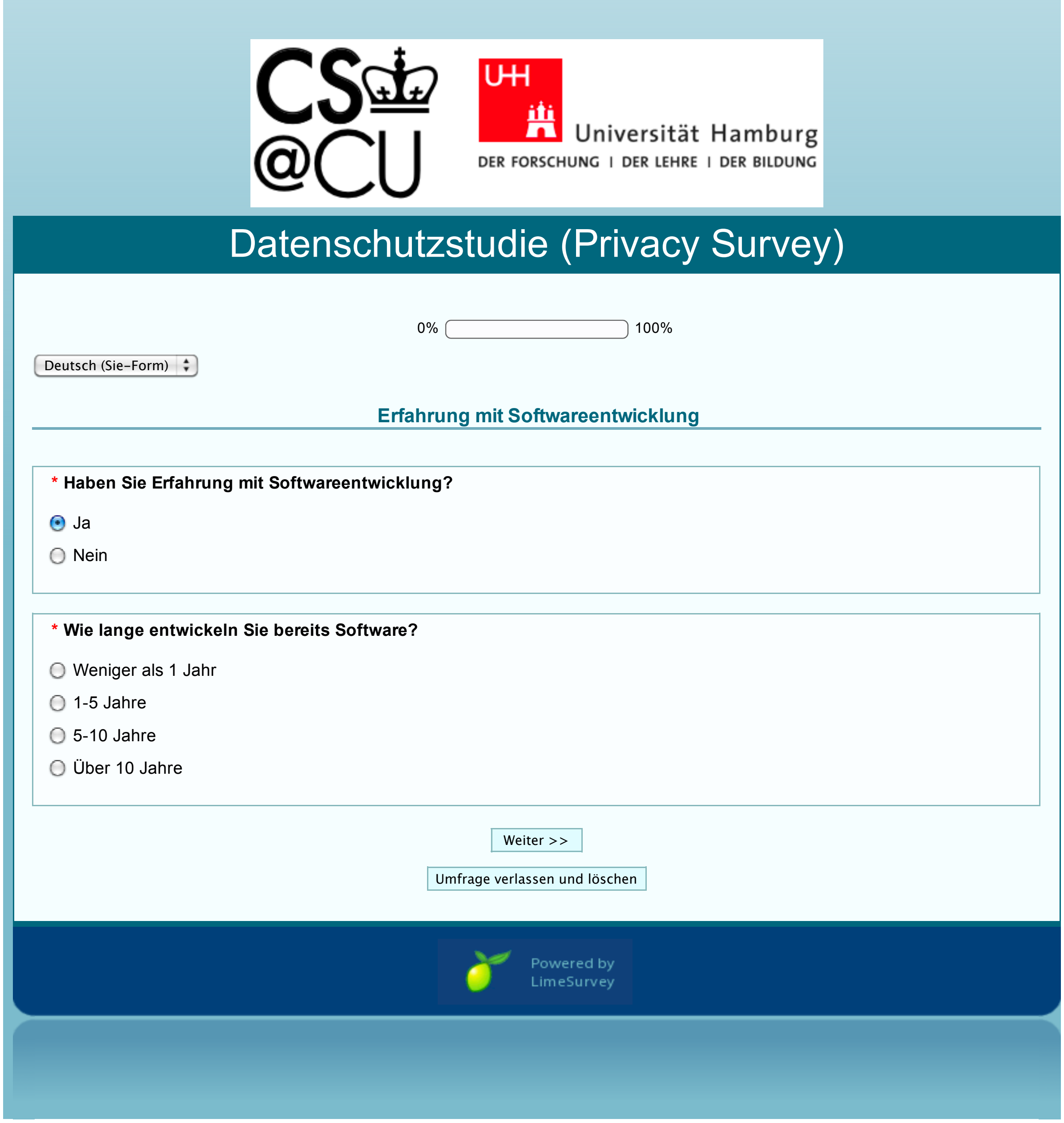
Unter den Teilnehmern dieser Studie werden zwei iPad Minis verlost.

Wenn Sie noch Fragen haben wenden Sie sich bitte an:

Swapneel Sheth (swapneel@cs.columbia.edu)

Walid Maalej (maalej@informatik.uni-hamburg.de)

* Diese Studie wurde durch das Columbia University’s Institutional Review Board mit der Nummer AAAJ8000 zugelassen.



Datenschutzstudie (Privacy Survey)

0% 100%

Erfahrung mit Softwareentwicklung

* **Haben Sie Erfahrung mit Softwareentwicklung?**

● Ja

○ Nein

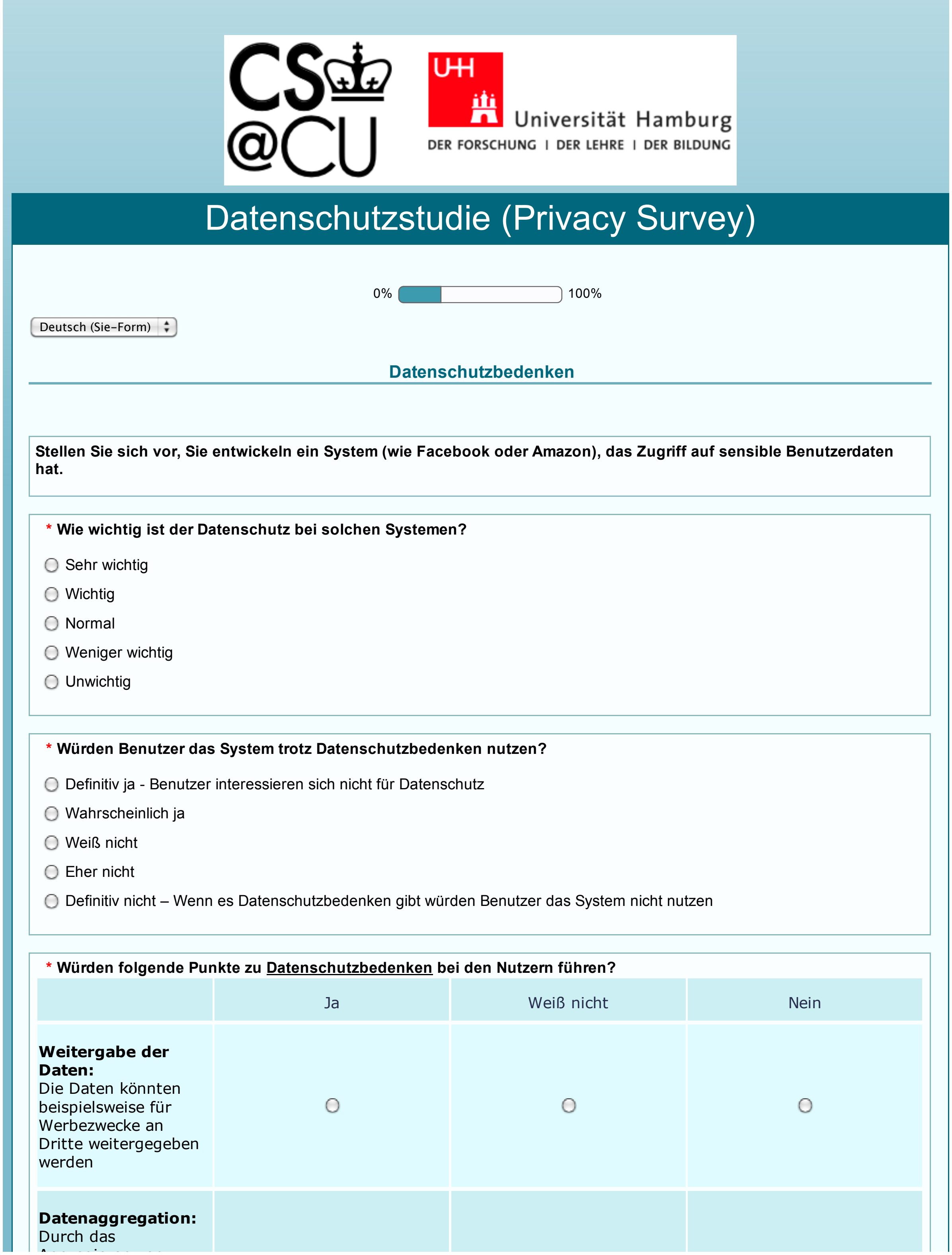
* **Wie lange entwickeln Sie bereits Software?**

○ Weniger als 1 Jahr

○ 1-5 Jahre

○ 5-10 Jahre

○ Über 10 Jahre



Datenschutzstudie (Privacy Survey)

Datenschutzbedenken

Stellen Sie sich vor, Sie entwickeln ein System (wie Facebook oder Amazon), das Zugriff auf sensible Benutzerdaten hat.

*** Wie wichtig ist der Datenschutz bei solchen Systemen?**

○ Sehr wichtig

○ Wichtig

○ Normal

○ Weniger wichtig

○ Unwichtig

*** Würden Benutzer das System trotz Datenschutzbedenken nutzen?**

○ Definitiv ja - Benutzer interessieren sich nicht für Datenschutz

○ Wahrscheinlich ja

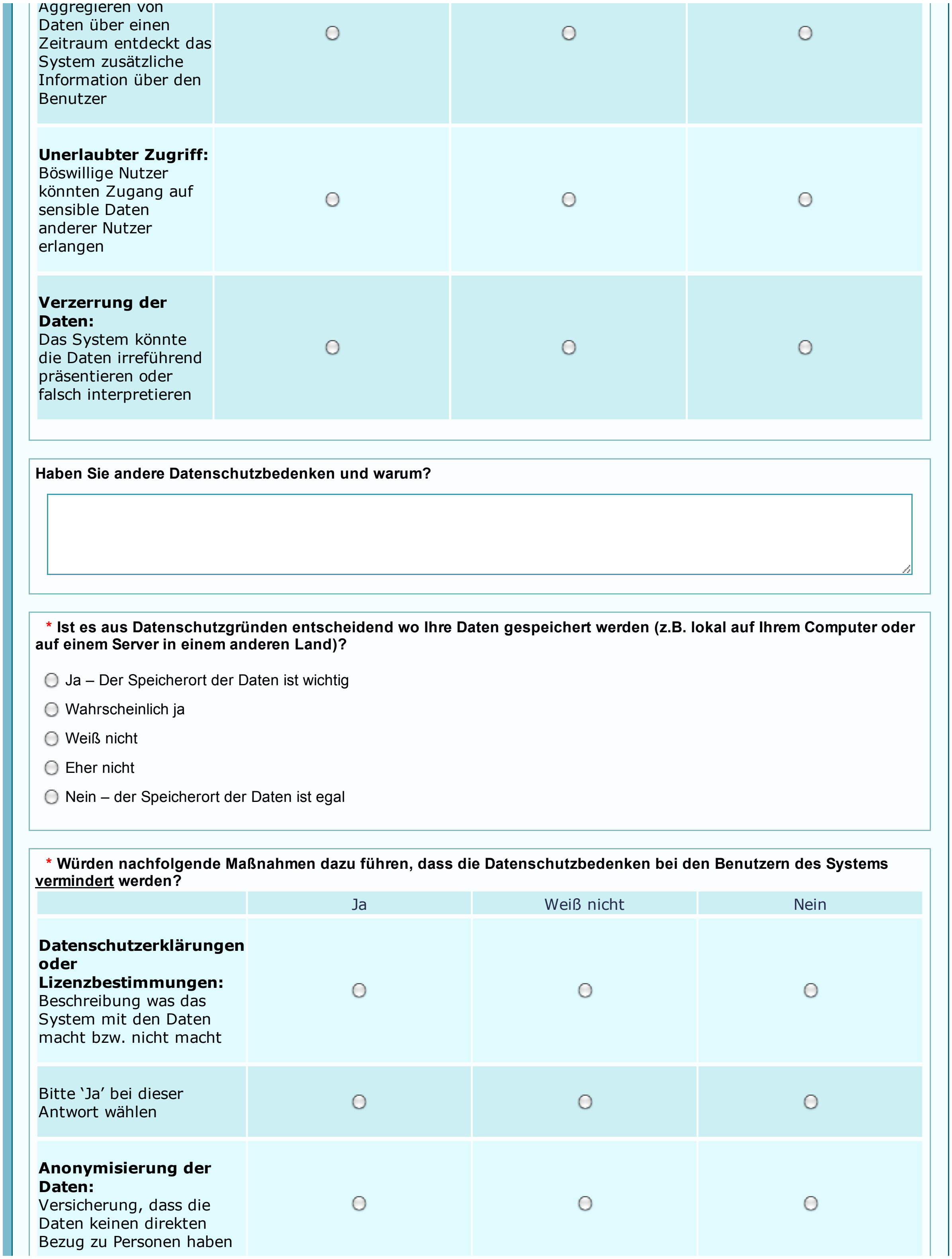
○ Weiß nicht

○ Eher nicht

○ Definitiv nicht – Wenn es Datenschutzbedenken gibt würden Benutzer das System nicht nutzen

*** Würden folgende Punkte zu Datenschutzbedenken bei den Nutzern führen?**

	Ja	Weiß nicht	Nein
Weitergabe der Daten: Die Daten könnten beispielsweise für Werbezwecke an Dritte weitergegeben werden	○	○	○
Datenaggregation: Durch das			



Aggregieren von Daten über einen Zeitraum entdeckt das System zusätzliche Information über den Benutzer	○	○	○
Unerlaubter Zugriff: Böswillige Nutzer könnten Zugang auf sensible Daten anderer Nutzer erlangen	○	○	○
Verzerrung der Daten: Das System könnte die Daten irreführend präsentieren oder falsch interpretieren	○	○	○

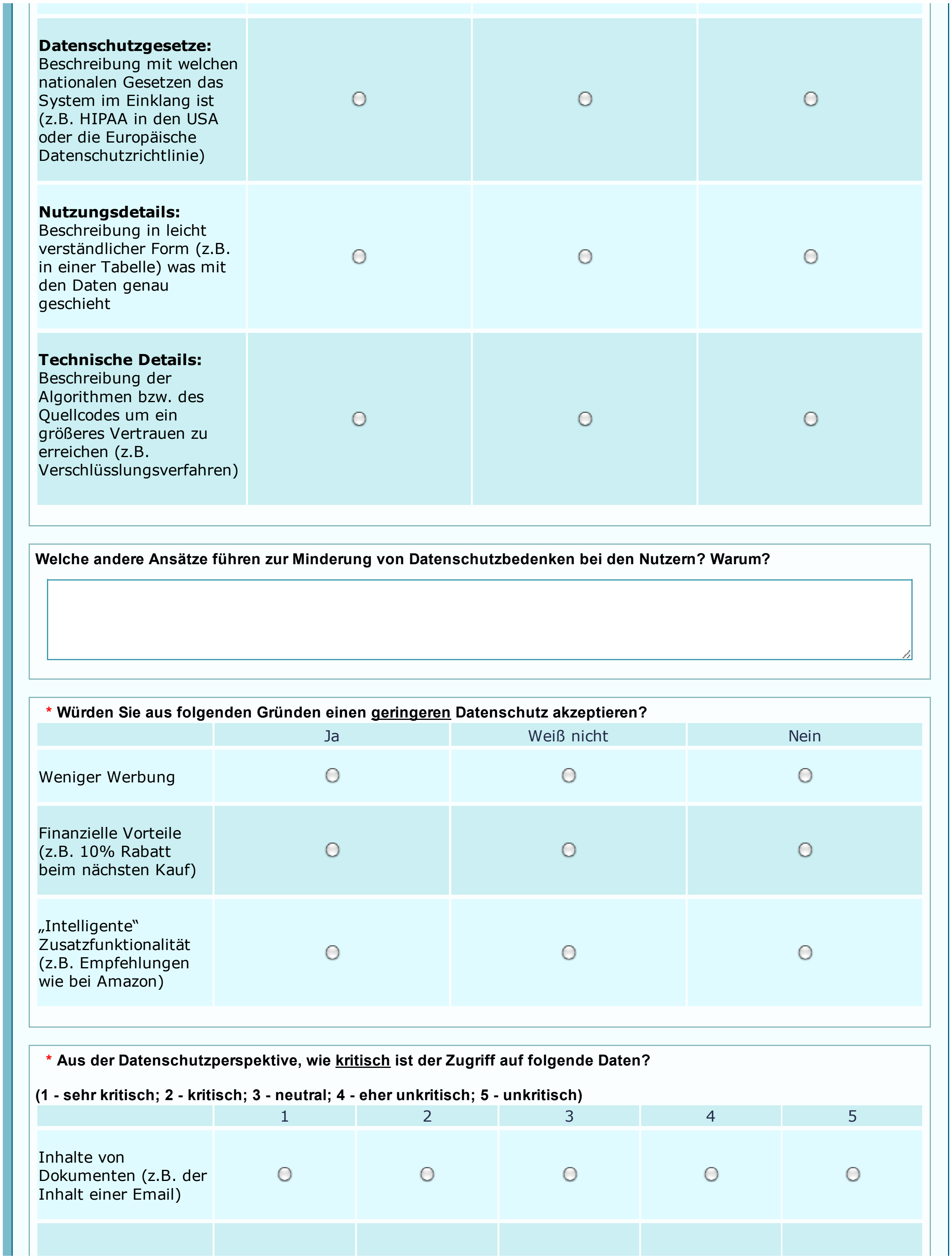
Haben Sie andere Datenschutzbedenken und warum?

*** Ist es aus Datenschutzgründen entscheidend wo Ihre Daten gespeichert werden (z.B. lokal auf Ihrem Computer oder auf einem Server in einem anderen Land)?**

- ○ Ja – Der Speicherort der Daten ist wichtig
- ○ Wahrscheinlich ja
- ○ Weiß nicht
- ○ Eher nicht
- ○ Nein – der Speicherort der Daten ist egal

*** Würden nachfolgende Maßnahmen dazu führen, dass die Datenschutzbedenken bei den Benutzern des Systems vermindert werden?**

	Ja	Weiß nicht	Nein
Datenschutzerklärungen oder Lizenzbestimmungen: Beschreibung was das System mit den Daten macht bzw. nicht macht	○	○	○
Bitte 'Ja' bei dieser Antwort wählen	○	○	○
Anonymisierung der Daten: Versicherung, dass die Daten keinen direkten Bezug zu Personen haben	○	○	○



Datenschutzgesetze: Beschreibung mit welchen nationalen Gesetzen das System im Einklang ist (z.B. HIPAA in den USA oder die Europäische Datenschutzrichtlinie)	○	○	○
Nutzungsdetails: Beschreibung in leicht verständlicher Form (z.B. in einer Tabelle) was mit den Daten genau geschieht	○	○	○
Technische Details: Beschreibung der Algorithmen bzw. des Quellcodes um ein größeres Vertrauen zu erreichen (z.B. Verschlüsslungsverfahren)	○	○	○

Welche andere Ansätze führen zur Minderung von Datenschutzbedenken bei den Nutzern? Warum?

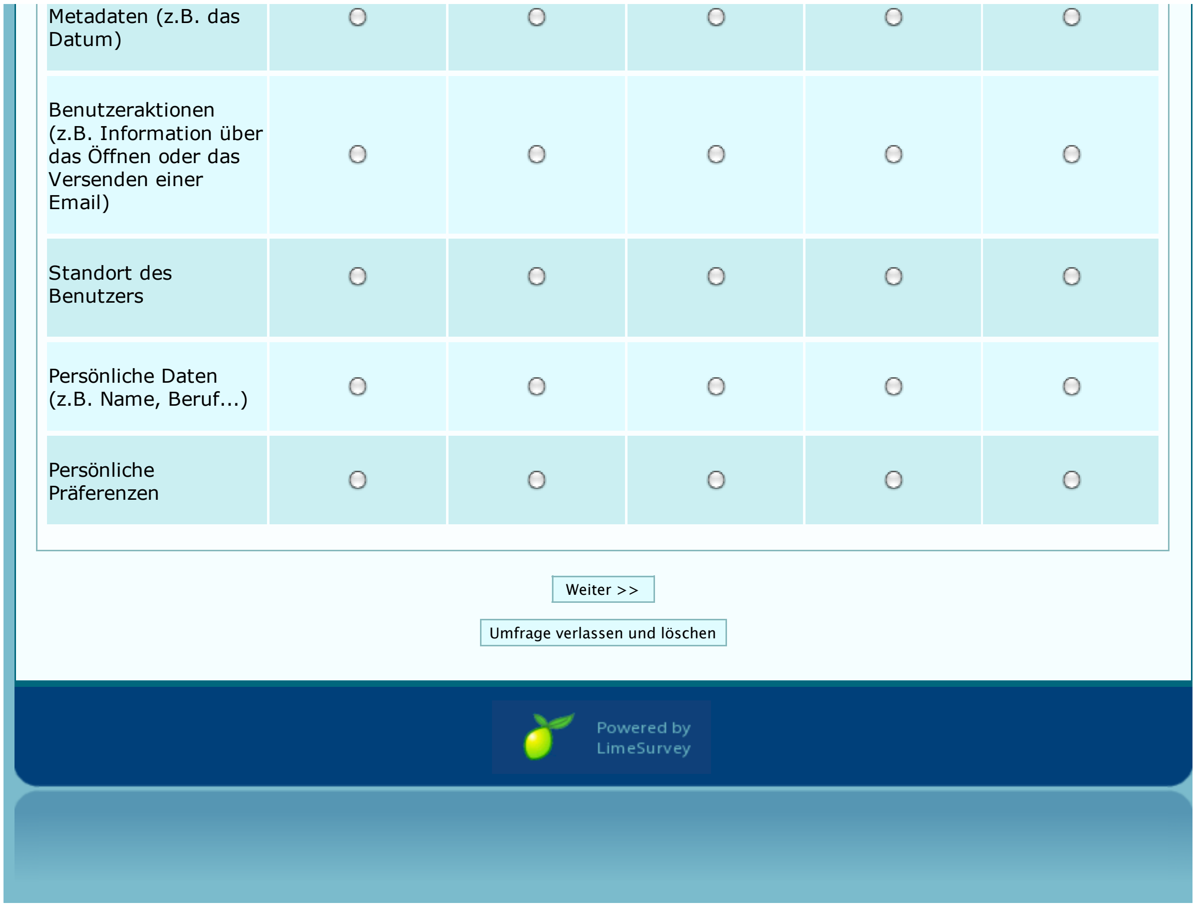
* **Würden Sie aus folgenden Gründen einen geringeren Datenschutz akzeptieren?**

	Ja	Weiß nicht	Nein
Weniger Werbung	○	○	○
Finanzielle Vorteile (z.B. 10% Rabatt beim nächsten Kauf)	○	○	○
„Intelligente" Zusatzfunktionalität (z.B. Empfehlungen wie bei Amazon)	○	○	○

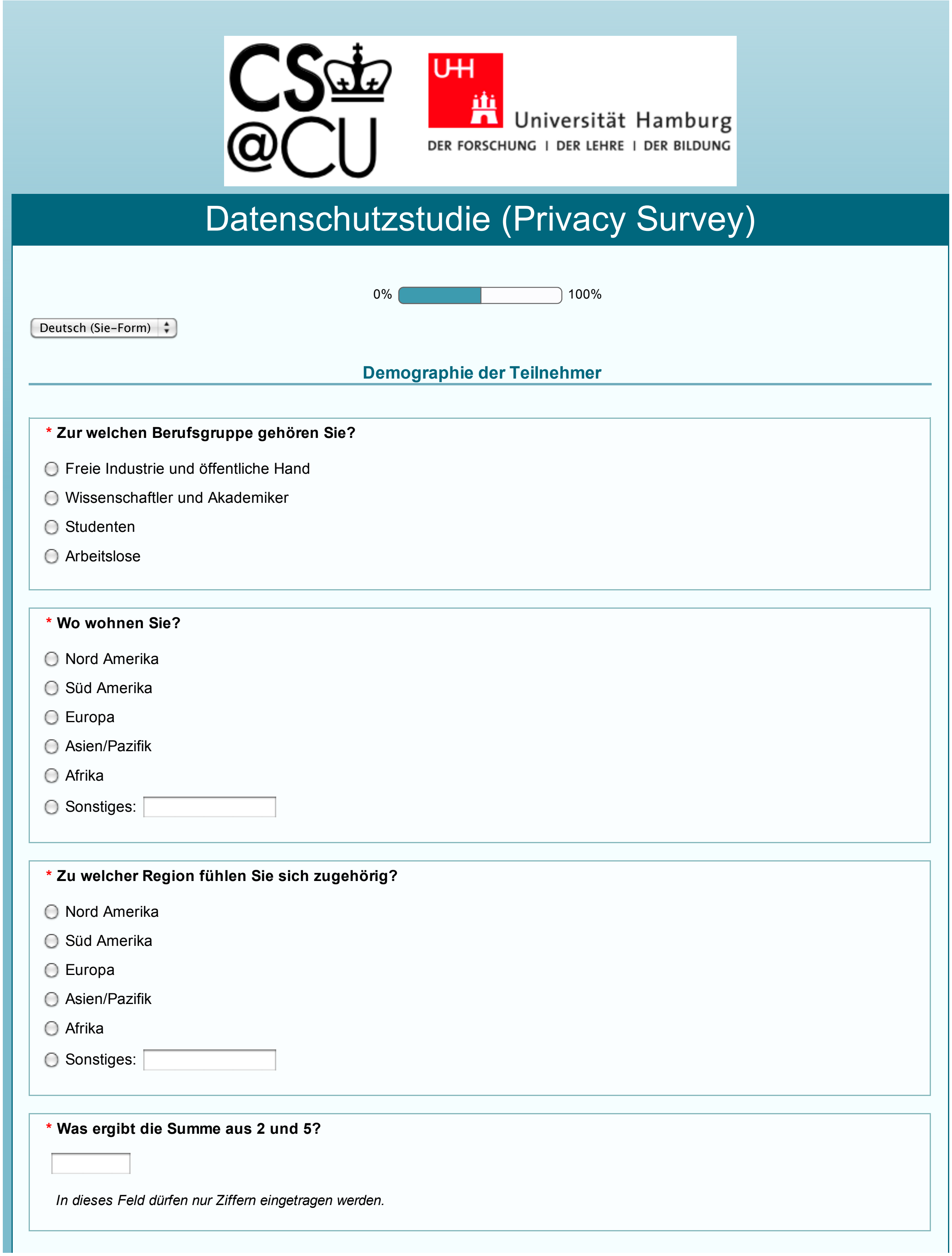
* **Aus der Datenschutzperspektive, wie kritisch ist der Zugriff auf folgende Daten?**

(1 - sehr kritisch; 2 - kritisch; 3 - neutral; 4 - eher unkritisch; 5 - unkritisch)

	1	2	3	4	5
Inhalte von Dokumenten (z.B. der Inhalt einer Email)	○	○	○	○	○



Metadaten (z.B. das Datum)	○	○	○	○	○
Benutzeraktionen (z.B. Information über das Öffnen oder das Versenden einer Email)	○	○	○	○	○
Standort des Benutzers	○	○	○	○	○
Persönliche Daten (z.B. Name, Beruf...)	○	○	○	○	○
Persönliche Präferenzen	○	○	○	○	○



Datenschutzstudie (Privacy Survey)

Demographie der Teilnehmer

* Zur welchen Berufsgruppe gehören Sie?

○ Freie Industrie und öffentliche Hand
○ Wissenschaftler und Akademiker
○ Studenten
○ Arbeitslose

* Wo wohnen Sie?

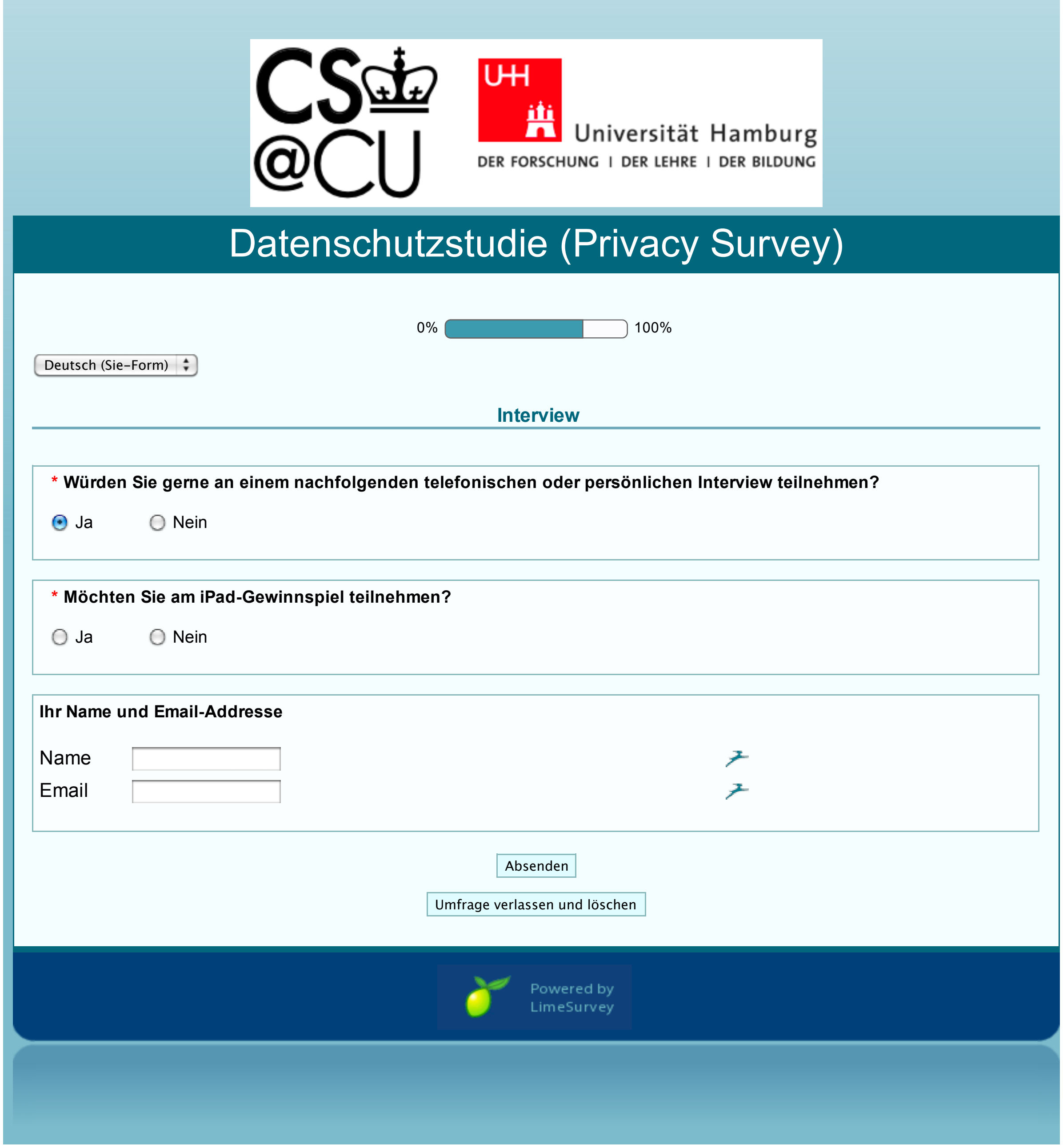
○ Nord Amerika
○ Süd Amerika
○ Europa
○ Asien/Pazifik
○ Afrika
○ Sonstiges:

* Zu welcher Region fühlen Sie sich zugehörig?

○ Nord Amerika
○ Süd Amerika
○ Europa
○ Asien/Pazifik
○ Afrika
○ Sonstiges:

* Was ergibt die Summe aus 2 und 5?

In dieses Feld dürfen nur Ziffern eingetragen werden.



Datenschutzstudie (Privacy Survey)

Interview

* Würden Sie gerne an einem nachfolgenden telefonischen oder persönlichen Interview teilnehmen?

● Ja ○ Nein

* Möchten Sie am iPad-Gewinnspiel teilnehmen?

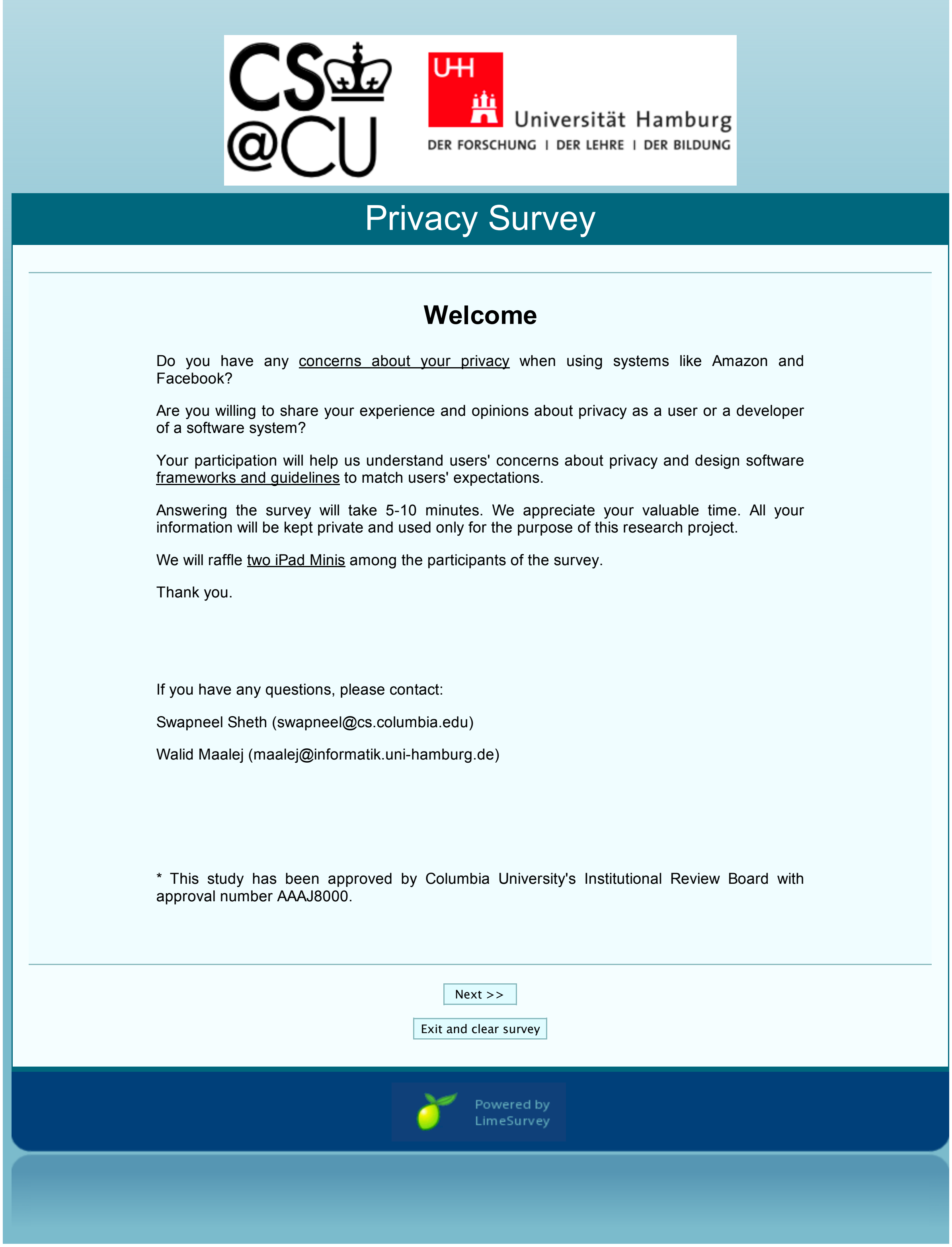
○ Ja ○ Nein

Ihr Name und Email-Addresse

Name

Email

A.3 Version for Users (English)



Privacy Survey

Welcome

Do you have any concerns about your privacy when using systems like Amazon and Facebook?

Are you willing to share your experience and opinions about privacy as a user or a developer of a software system?

Your participation will help us understand users' concerns about privacy and design software frameworks and guidelines to match users' expectations.

Answering the survey will take 5-10 minutes. We appreciate your valuable time. All your information will be kept private and used only for the purpose of this research project.

We will raffle two iPad Minis among the participants of the survey.

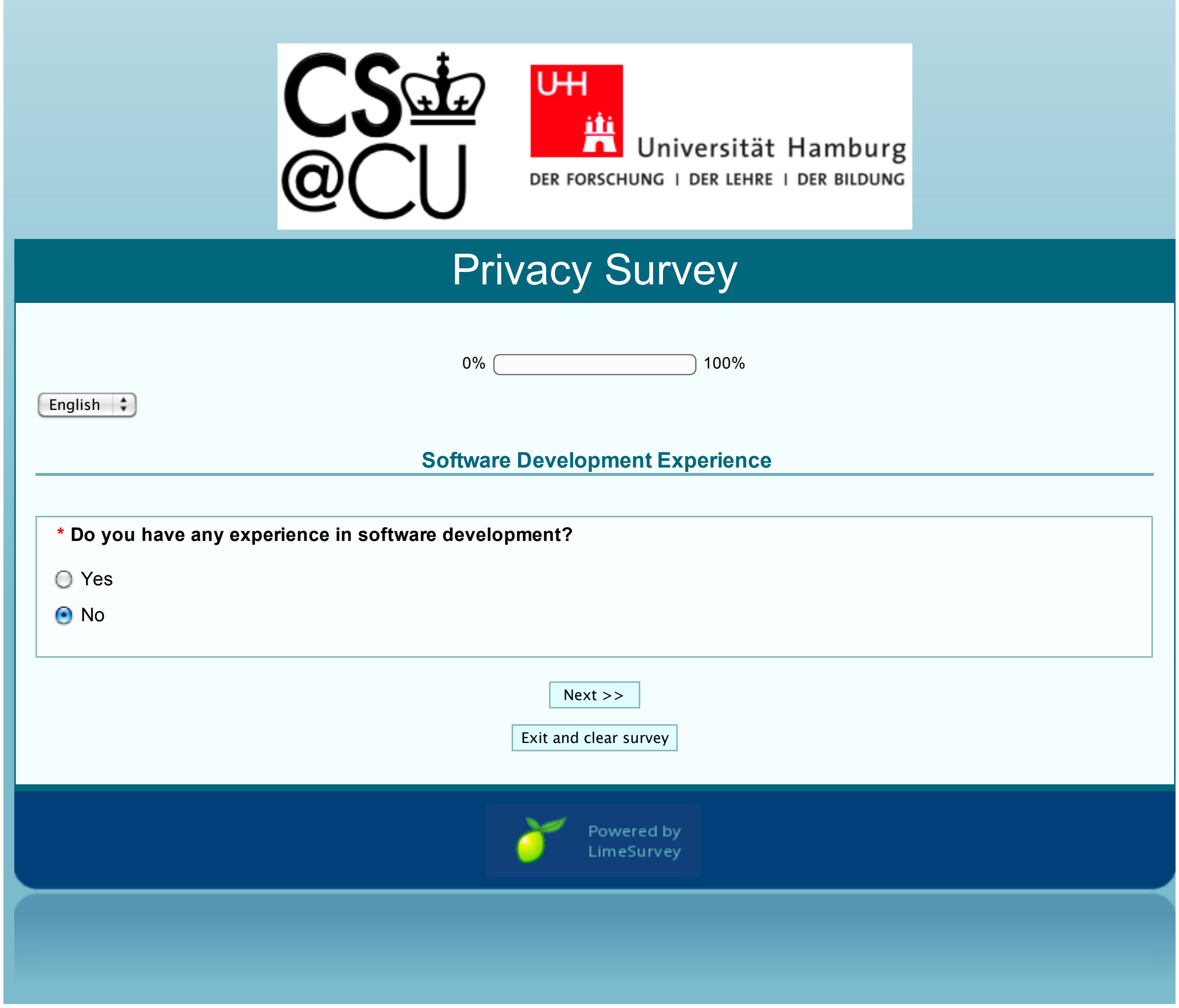
Thank you.

If you have any questions, please contact:

Swapneel Sheth (swapneel@cs.columbia.edu)

Walid Maalej (maalej@informatik.uni-hamburg.de)

* This study has been approved by Columbia University's Institutional Review Board with approval number AAAJ8000.



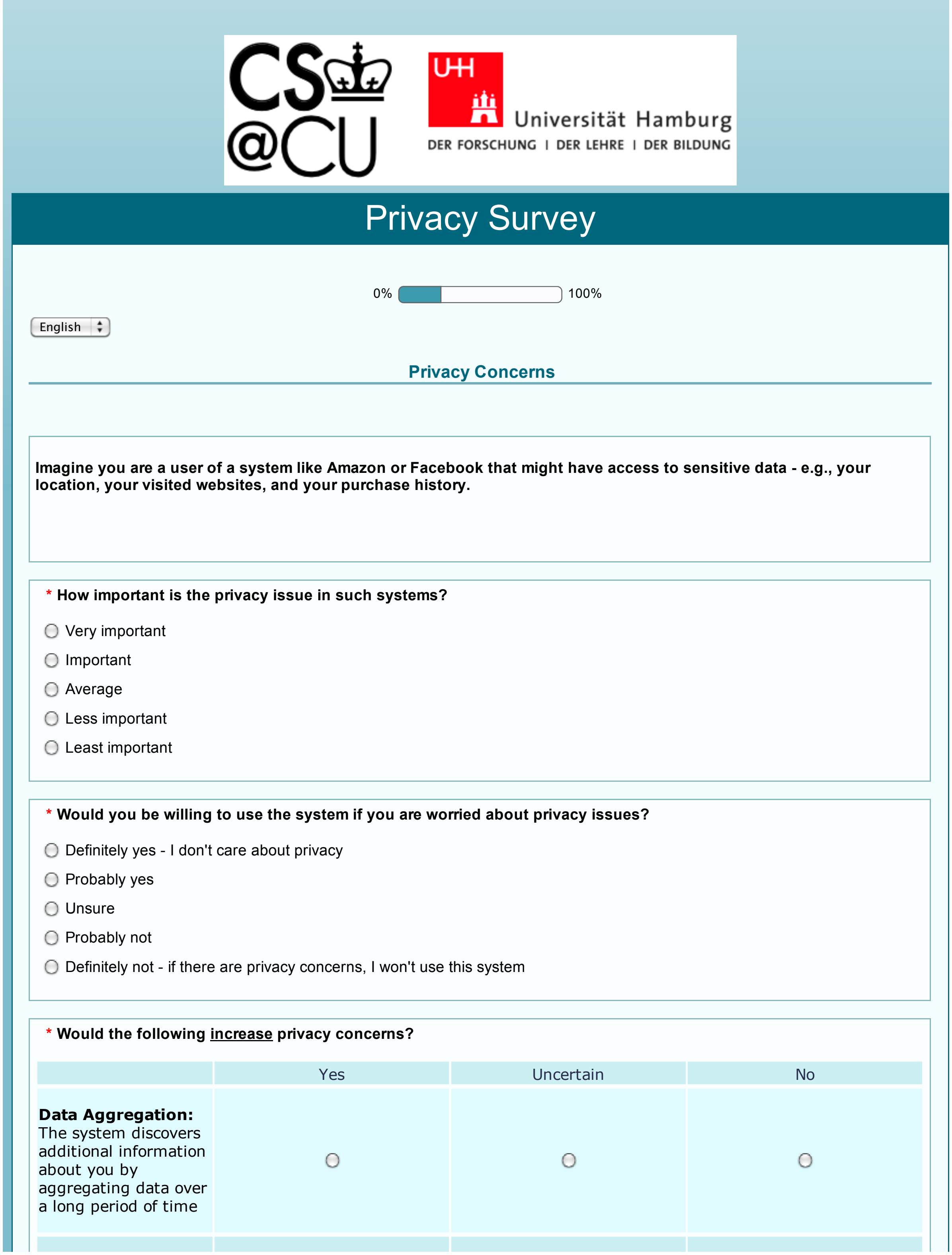
Privacy Survey

Software Development Experience

* Do you have any experience in software development?

○ Yes

● No



Privacy Survey

Privacy Concerns

Imagine you are a user of a system like Amazon or Facebook that might have access to sensitive data - e.g., your location, your visited websites, and your purchase history.

* **How important is the privacy issue in such systems?**

○ Very important

○ Important

○ Average

○ Less important

○ Least important

* **Would you be willing to use the system if you are worried about privacy issues?**

○ Definitely yes - I don't care about privacy

○ Probably yes

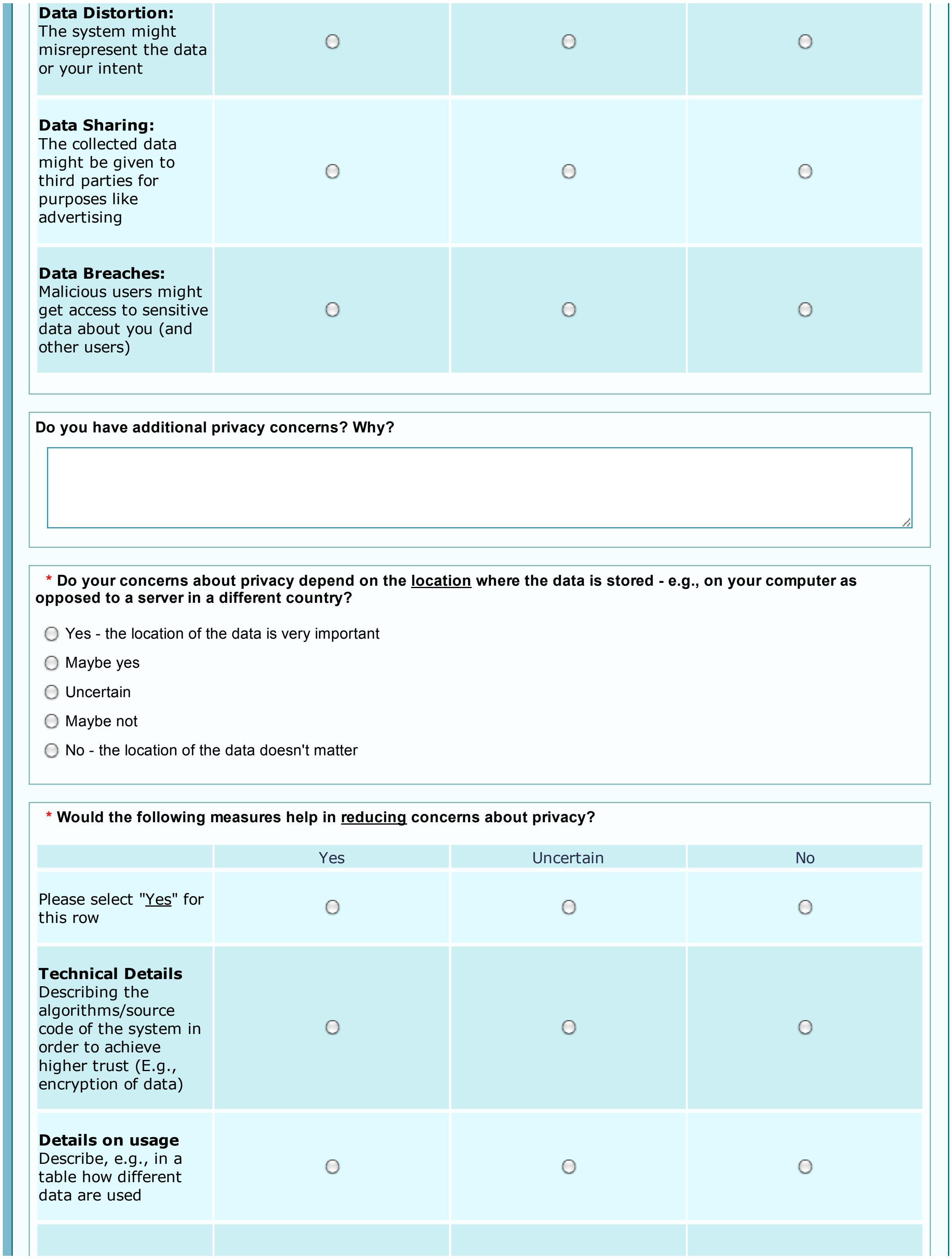
○ Unsure

○ Probably not

○ Definitely not - if there are privacy concerns, I won't use this system

* **Would the following increase privacy concerns?**

	Yes	Uncertain	No
Data Aggregation: The system discovers additional information about you by aggregating data over a long period of time	○	○	○



Data Distortion: The system might misrepresent the data or your intent	○	○	○
Data Sharing: The collected data might be given to third parties for purposes like advertising	○	○	○
Data Breaches: Malicious users might get access to sensitive data about you (and other users)	○	○	○

Do you have additional privacy concerns? Why?

* **Do your concerns about privacy depend on the location where the data is stored - e.g., on your computer as opposed to a server in a different country?**

○ Yes - the location of the data is very important

○ Maybe yes

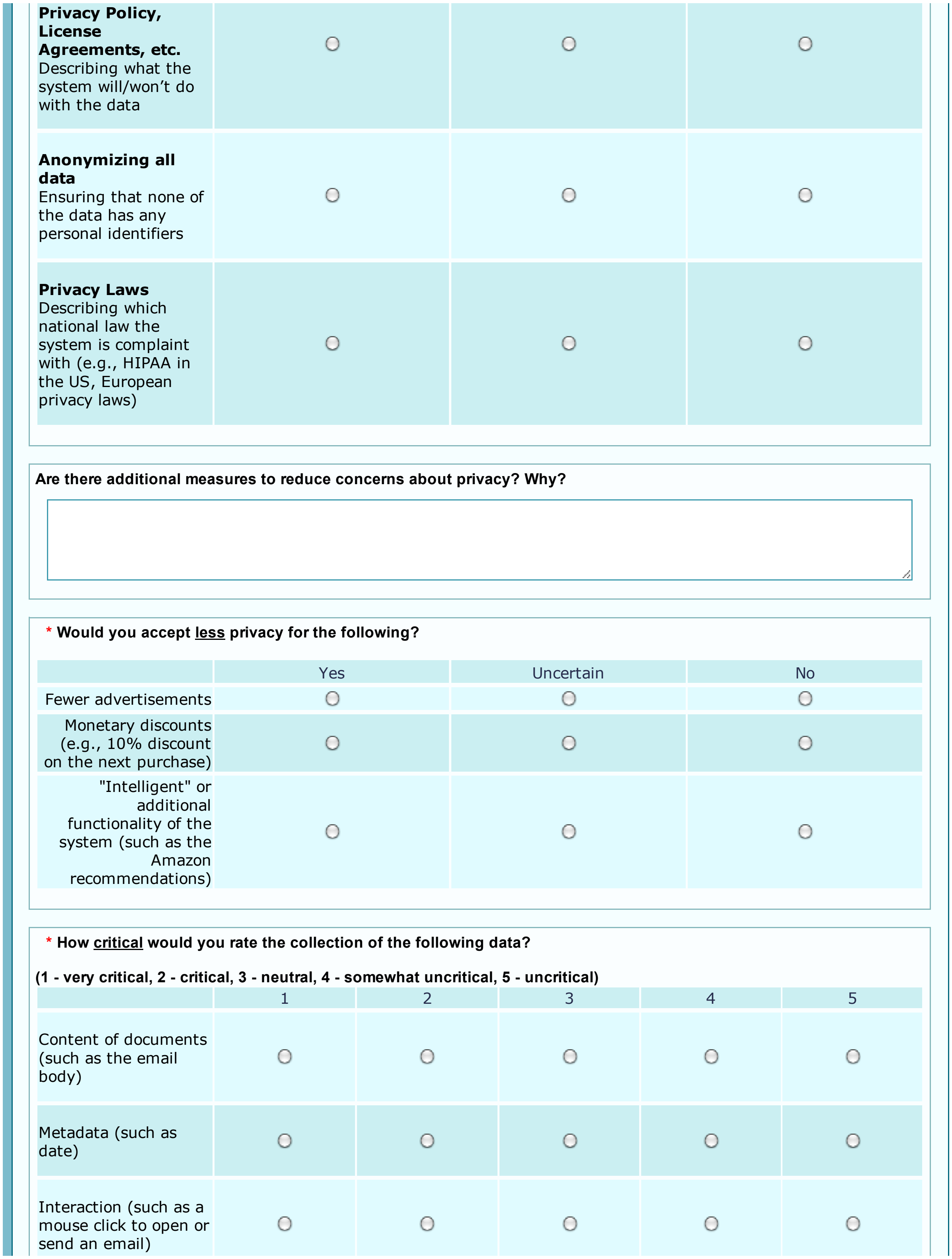
○ Uncertain

○ Maybe not

○ No - the location of the data doesn't matter

* **Would the following measures help in reducing concerns about privacy?**

	Yes	Uncertain	No
Please select "Yes" for this row	○	○	○
Technical Details Describing the algorithms/source code of the system in order to achieve higher trust (E.g., encryption of data)	○	○	○
Details on usage Describe, e.g., in a table how different data are used	○	○	○



Privacy Policy, License Agreements, etc. Describing what the system will/won't do with the data	○	○	○
Anonymizing all data Ensuring that none of the data has any personal identifiers	○	○	○
Privacy Laws Describing which national law the system is complaint with (e.g., HIPAA in the US, European privacy laws)	○	○	○

Are there additional measures to reduce concerns about privacy? Why?

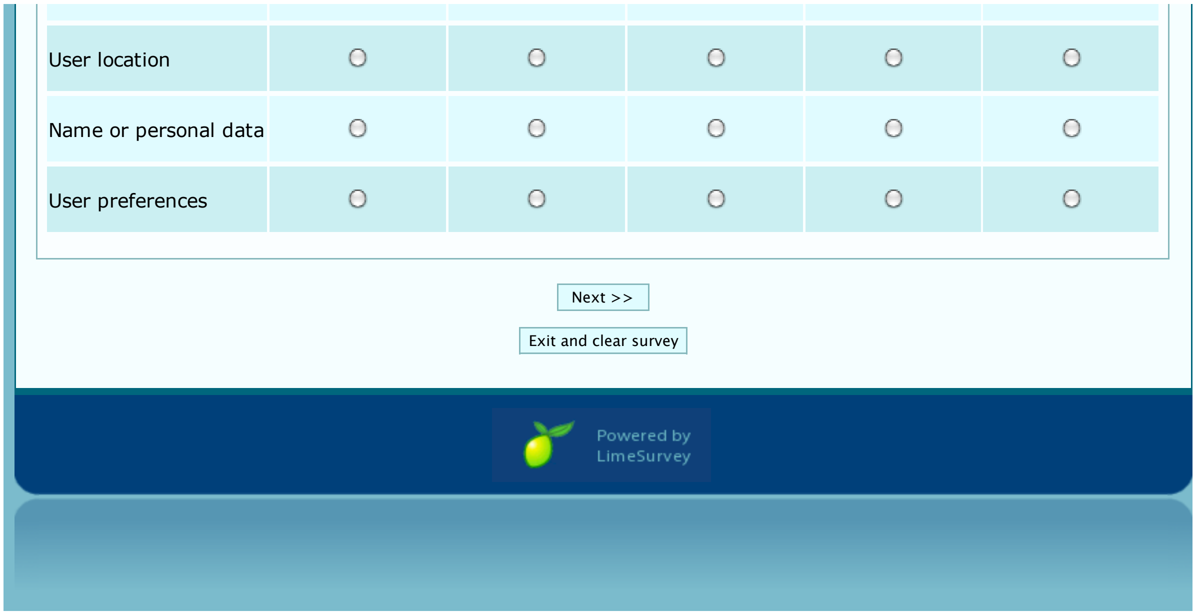
* **Would you accept less privacy for the following?**

	Yes	Uncertain	No
Fewer advertisements	○	○	○
Monetary discounts (e.g., 10% discount on the next purchase)	○	○	○
"Intelligent" or additional functionality of the system (such as the Amazon recommendations)	○	○	○

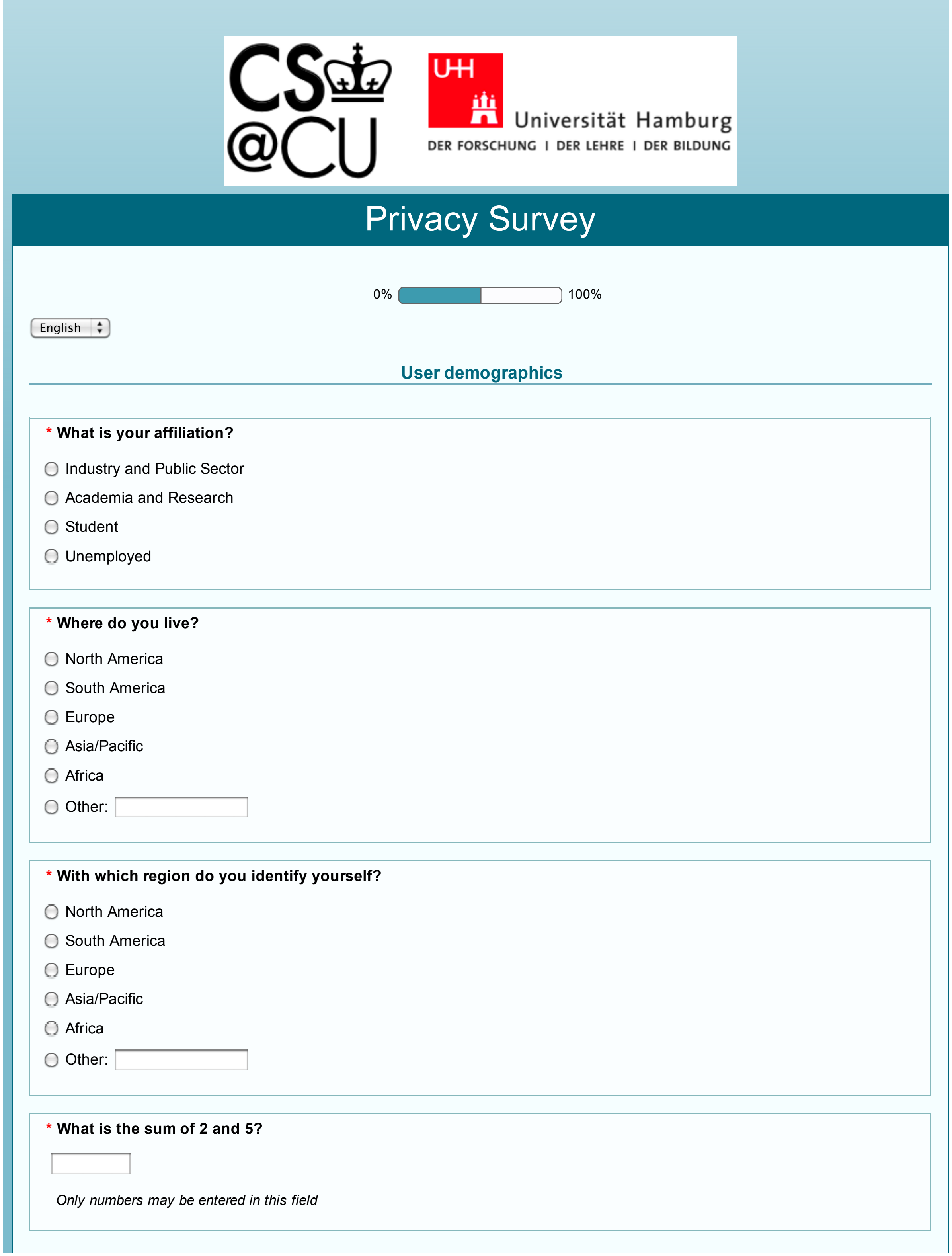
* **How critical would you rate the collection of the following data?**

(1 - very critical, 2 - critical, 3 - neutral, 4 - somewhat uncritical, 5 - uncritical)

	1	2	3	4	5
Content of documents (such as the email body)	○	○	○	○	○
Metadata (such as date)	○	○	○	○	○
Interaction (such as a mouse click to open or send an email)	○	○	○	○	○



User location	○	○	○	○	○
Name or personal data	○	○	○	○	○
User preferences	○	○	○	○	○



Privacy Survey

0% 100%

English

User demographics

* What is your affiliation?

○ Industry and Public Sector
○ Academia and Research
○ Student
○ Unemployed

* Where do you live?

○ North America
○ South America
○ Europe
○ Asia/Pacific
○ Africa
○ Other: ____

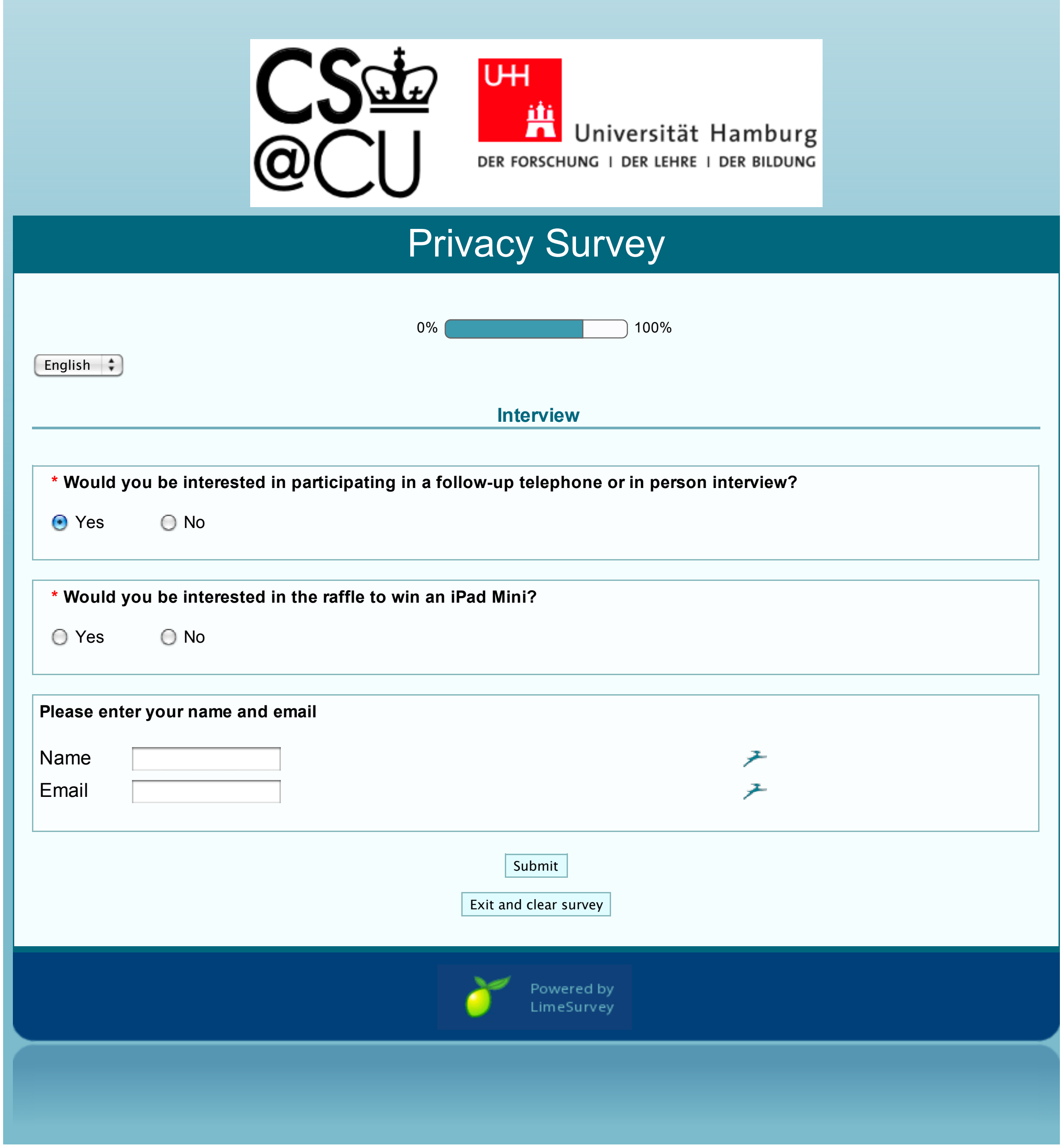
* With which region do you identify yourself?

○ North America
○ South America
○ Europe
○ Asia/Pacific
○ Africa
○ Other: ____

* What is the sum of 2 and 5?

____

Only numbers may be entered in this field



Privacy Survey

Interview

* Would you be interested in participating in a follow-up telephone or in person interview?

● Yes ○ No

* Would you be interested in the raffle to win an iPad Mini?

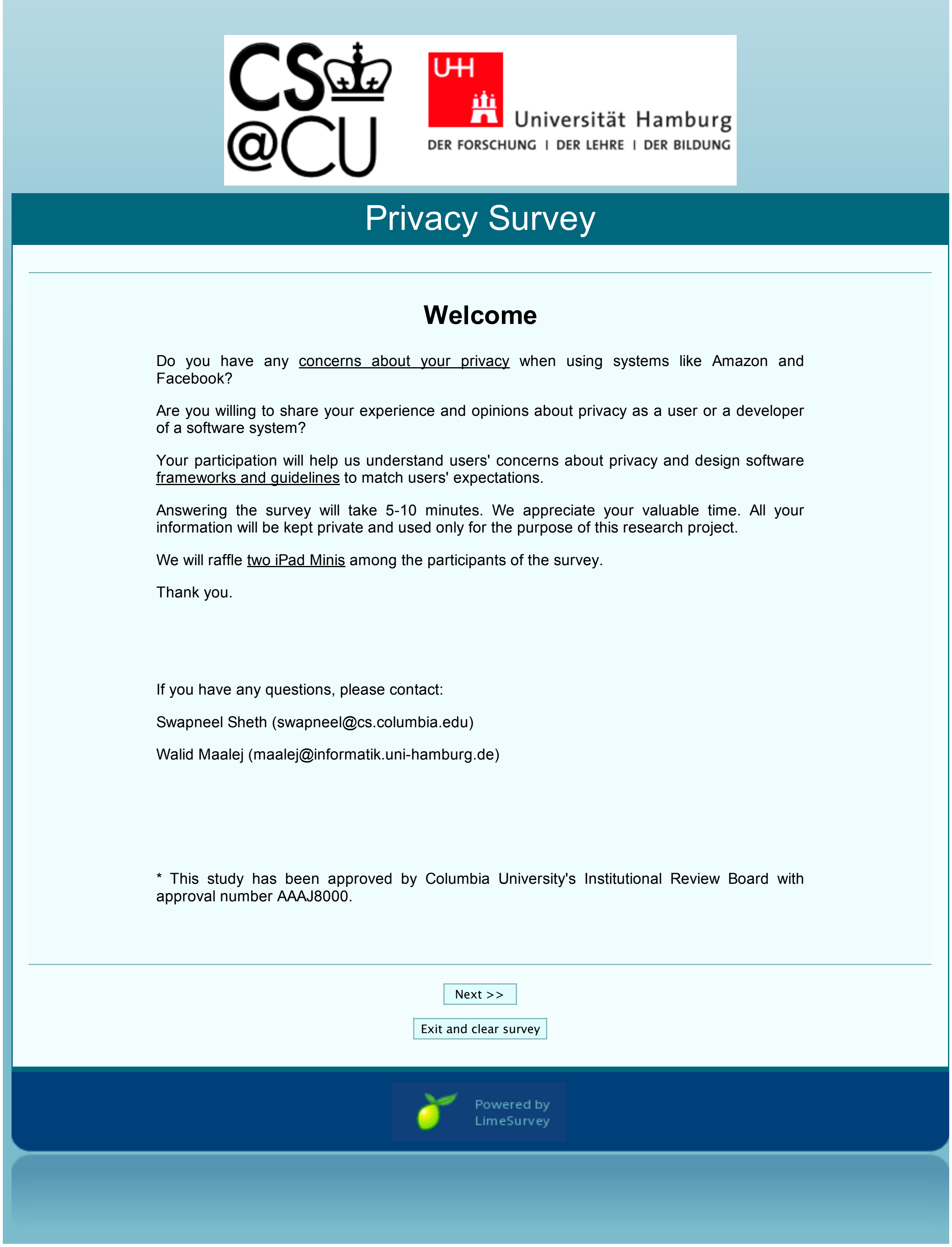
○ Yes ○ No

Please enter your name and email

Name

Email

A.4 Version for Users (German)



Privacy Survey

Welcome

Do you have any concerns about your privacy when using systems like Amazon and Facebook?

Are you willing to share your experience and opinions about privacy as a user or a developer of a software system?

Your participation will help us understand users' concerns about privacy and design software frameworks and guidelines to match users' expectations.

Answering the survey will take 5-10 minutes. We appreciate your valuable time. All your information will be kept private and used only for the purpose of this research project.

We will raffle two iPad Minis among the participants of the survey.

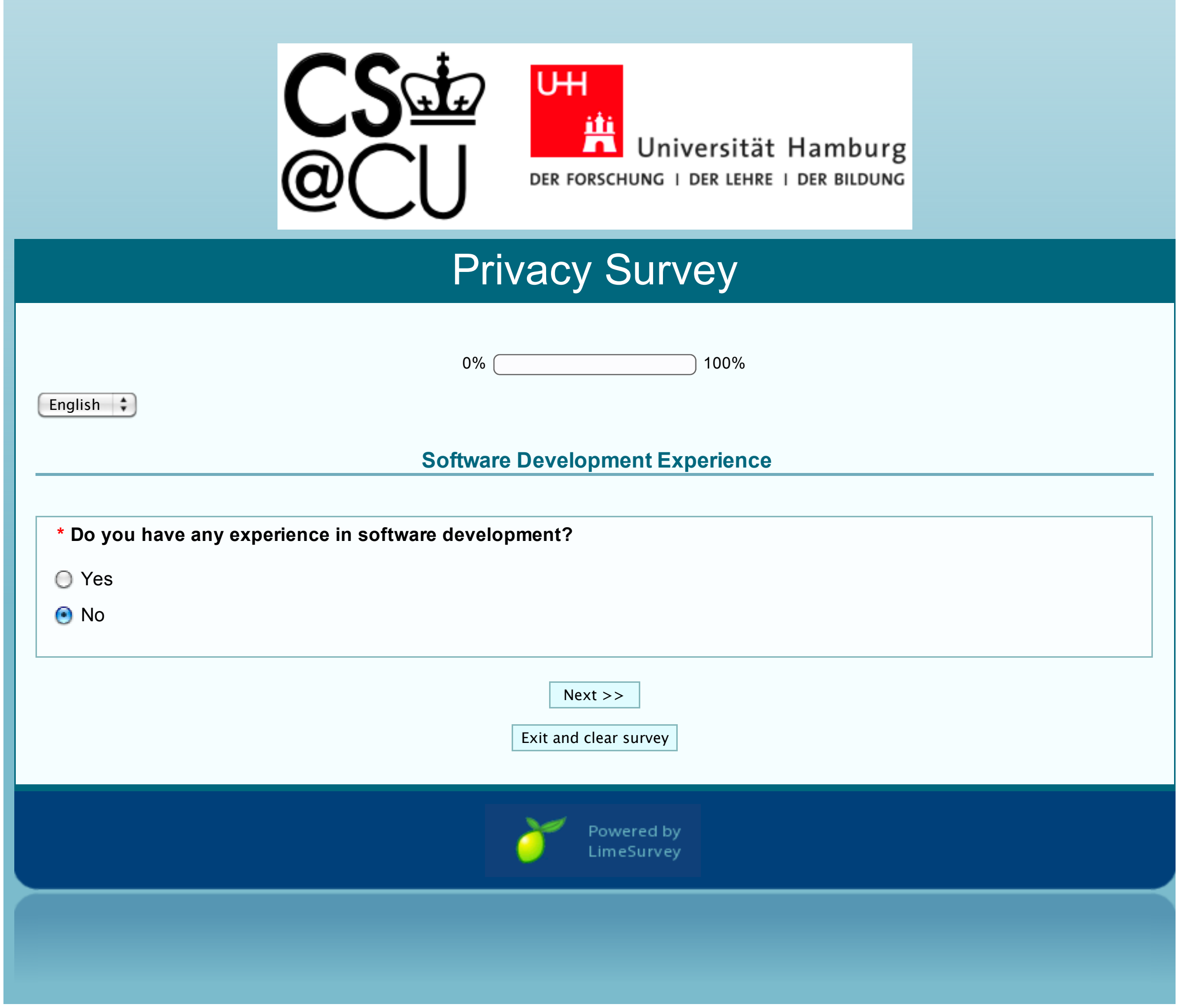
Thank you.

If you have any questions, please contact:

Swapneel Sheth (swapneel@cs.columbia.edu)

Walid Maalej (maalej@informatik.uni-hamburg.de)

* This study has been approved by Columbia University's Institutional Review Board with approval number AAAJ8000.



Privacy Survey

0% 100%

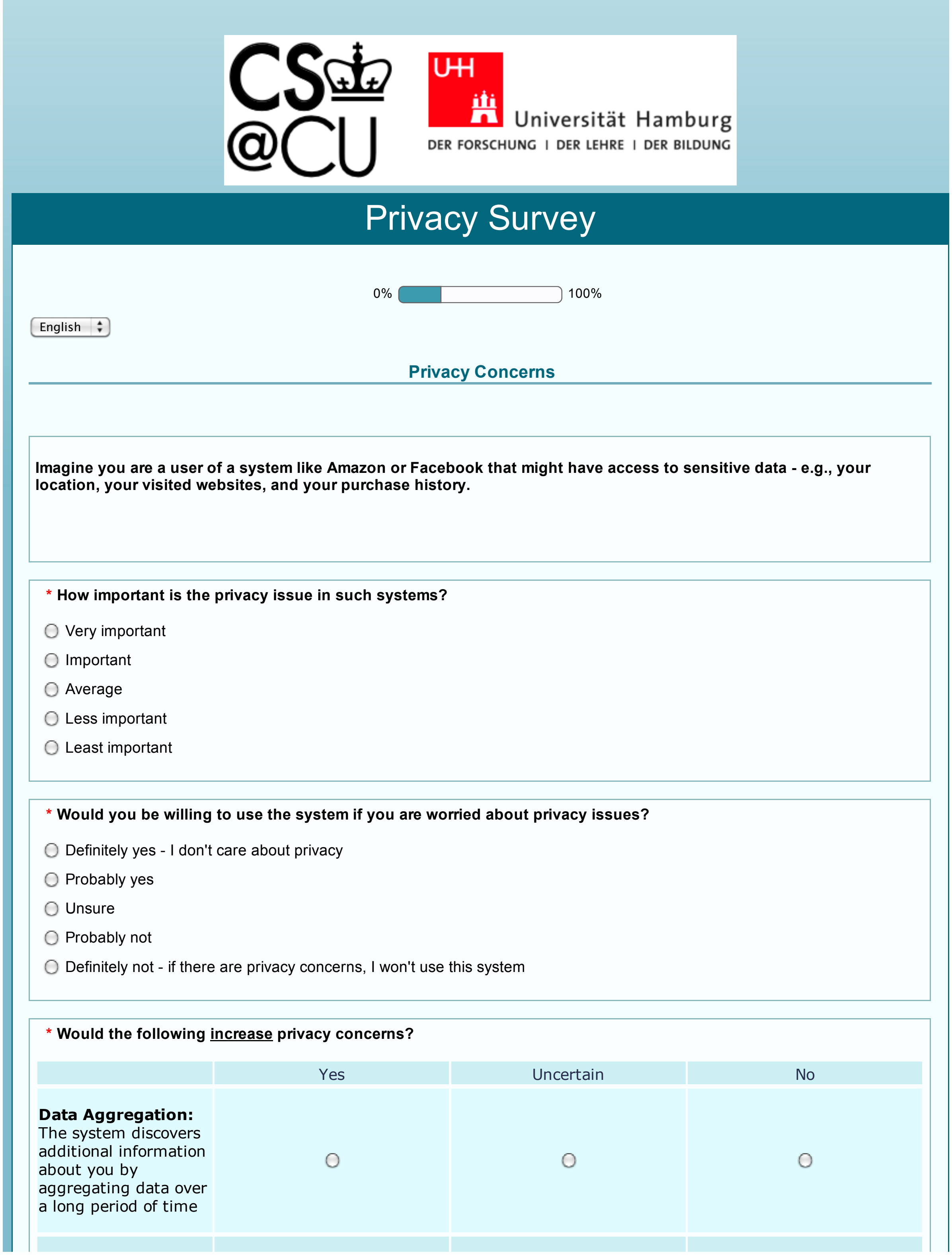
English

Software Development Experience

* Do you have any experience in software development?

○ Yes

● No



Privacy Survey

Privacy Concerns

Imagine you are a user of a system like Amazon or Facebook that might have access to sensitive data - e.g., your location, your visited websites, and your purchase history.

* **How important is the privacy issue in such systems?**

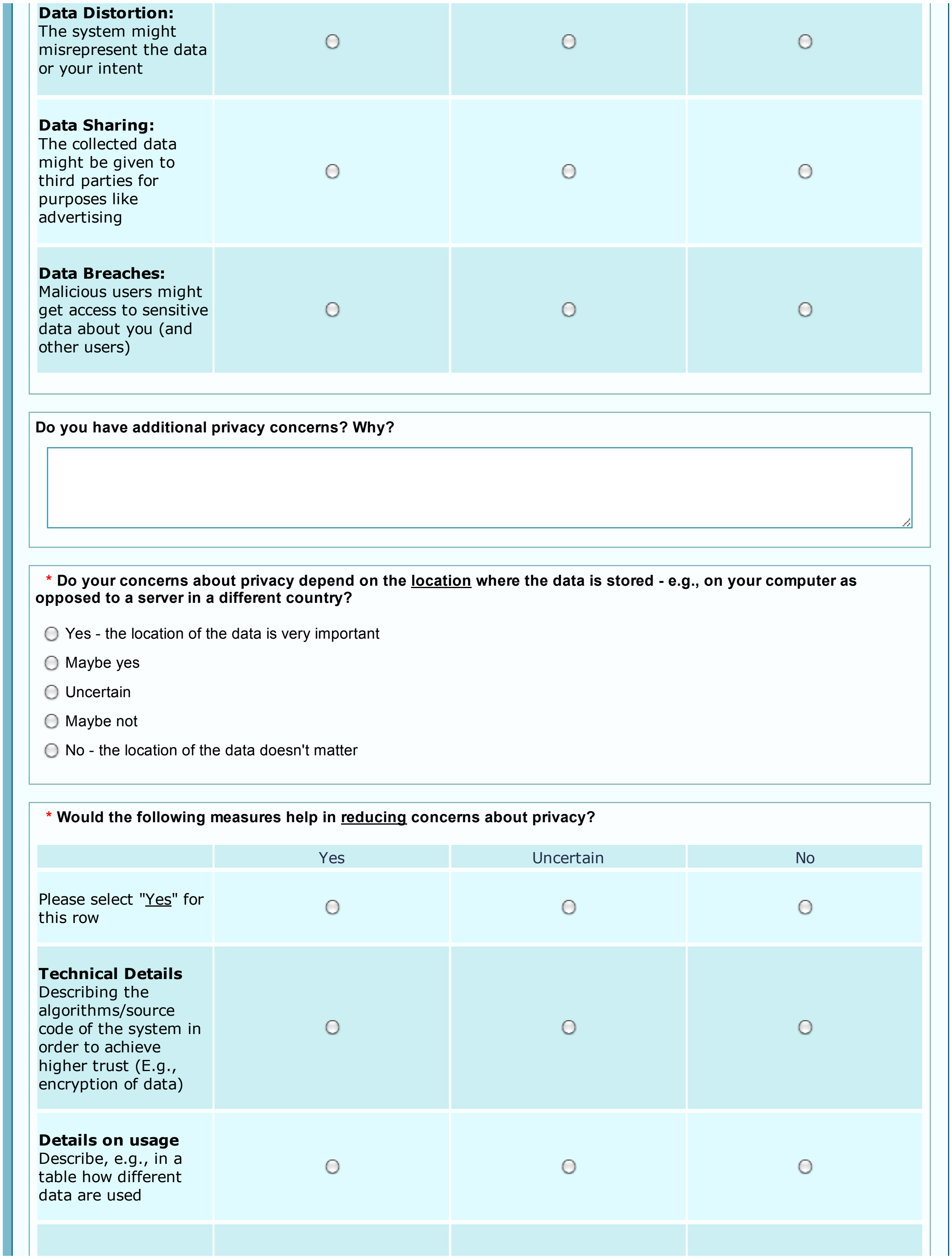
○ Very important
○ Important
○ Average
○ Less important
○ Least important

* **Would you be willing to use the system if you are worried about privacy issues?**

○ Definitely yes - I don't care about privacy
○ Probably yes
○ Unsure
○ Probably not
○ Definitely not - if there are privacy concerns, I won't use this system

* **Would the following increase privacy concerns?**

	Yes	Uncertain	No
Data Aggregation: The system discovers additional information about you by aggregating data over a long period of time	○	○	○



Data Distortion: The system might misrepresent the data or your intent	○	○	○
Data Sharing: The collected data might be given to third parties for purposes like advertising	○	○	○
Data Breaches: Malicious users might get access to sensitive data about you (and other users)	○	○	○

Do you have additional privacy concerns? Why?

* **Do your concerns about privacy depend on the location where the data is stored - e.g., on your computer as opposed to a server in a different country?**

○ Yes - the location of the data is very important

○ Maybe yes

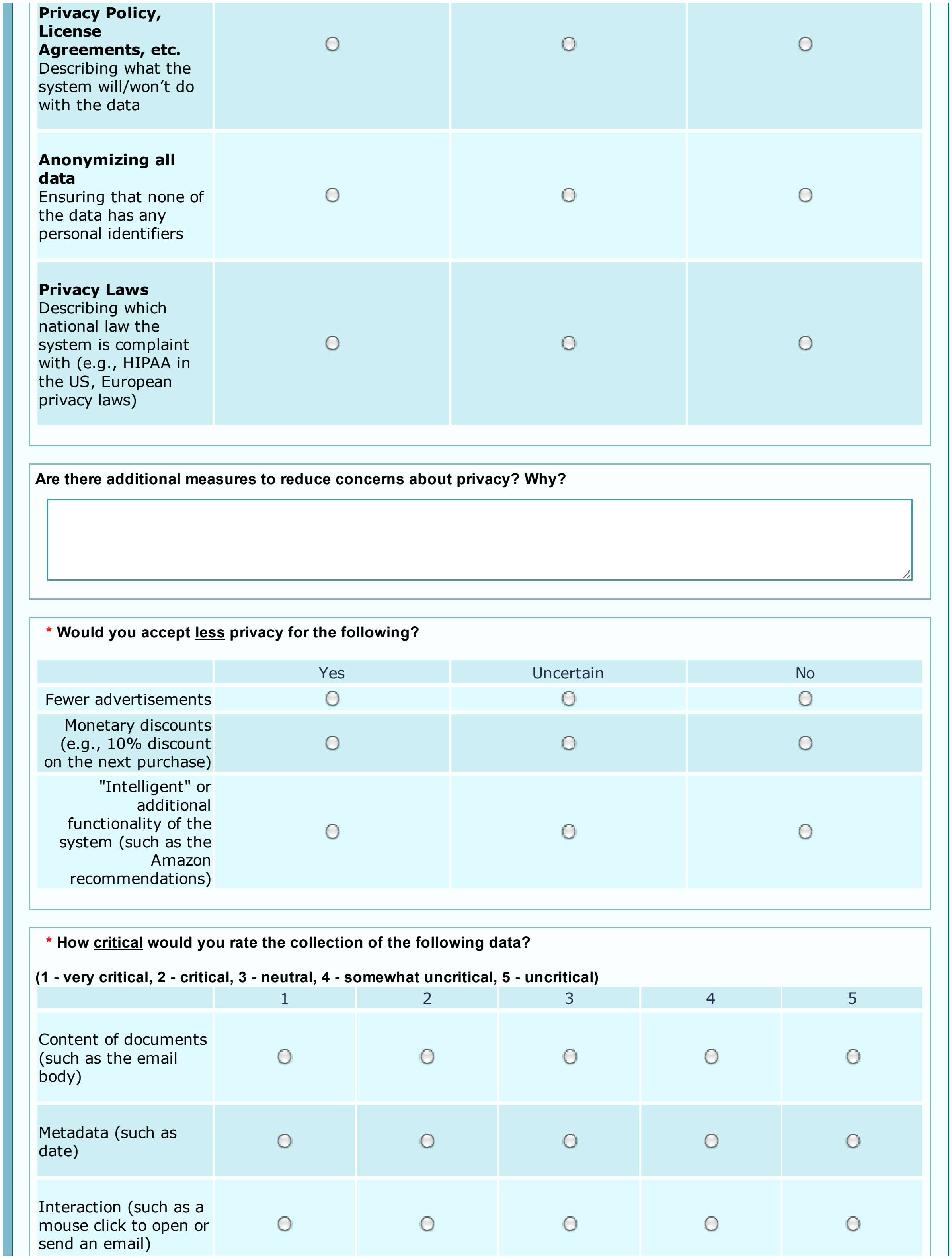
○ Uncertain

○ Maybe not

○ No - the location of the data doesn't matter

* **Would the following measures help in reducing concerns about privacy?**

	Yes	Uncertain	No
Please select "Yes" for this row	○	○	○
Technical Details Describing the algorithms/source code of the system in order to achieve higher trust (E.g., encryption of data)	○	○	○
Details on usage Describe, e.g., in a table how different data are used	○	○	○



Privacy Policy, License Agreements, etc. Describing what the system will/won't do with the data	○	○	○
Anonymizing all data Ensuring that none of the data has any personal identifiers	○	○	○
Privacy Laws Describing which national law the system is complaint with (e.g., HIPAA in the US, European privacy laws)	○	○	○

Are there additional measures to reduce concerns about privacy? Why?

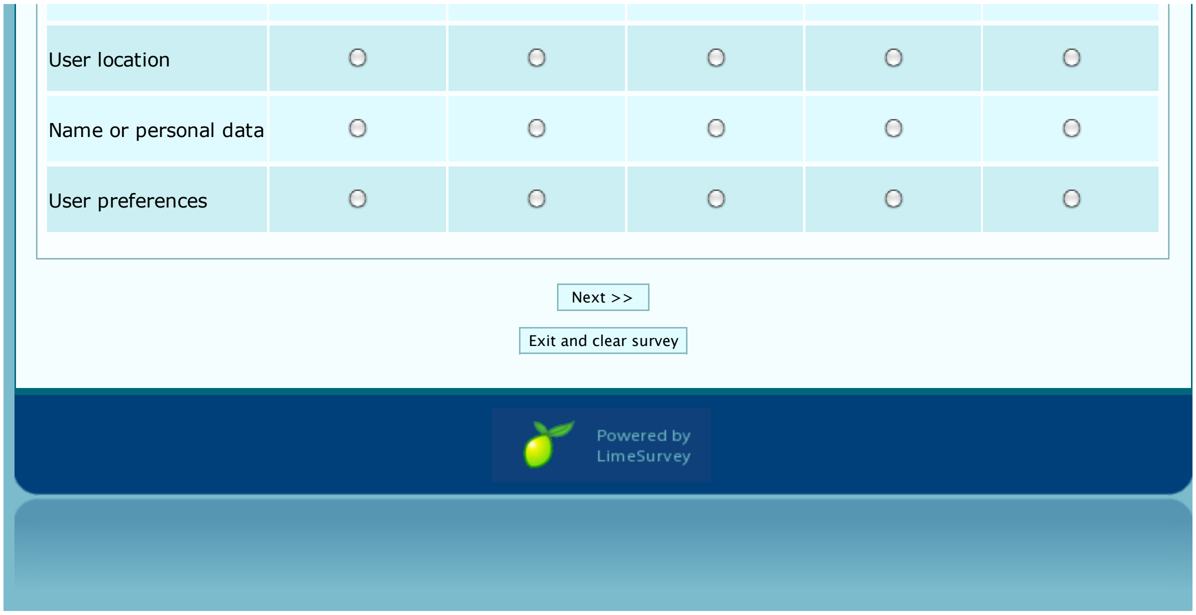
*** Would you accept less privacy for the following?**

	Yes	Uncertain	No
Fewer advertisements	○	○	○
Monetary discounts (e.g., 10% discount on the next purchase)	○	○	○
"Intelligent" or additional functionality of the system (such as the Amazon recommendations)	○	○	○

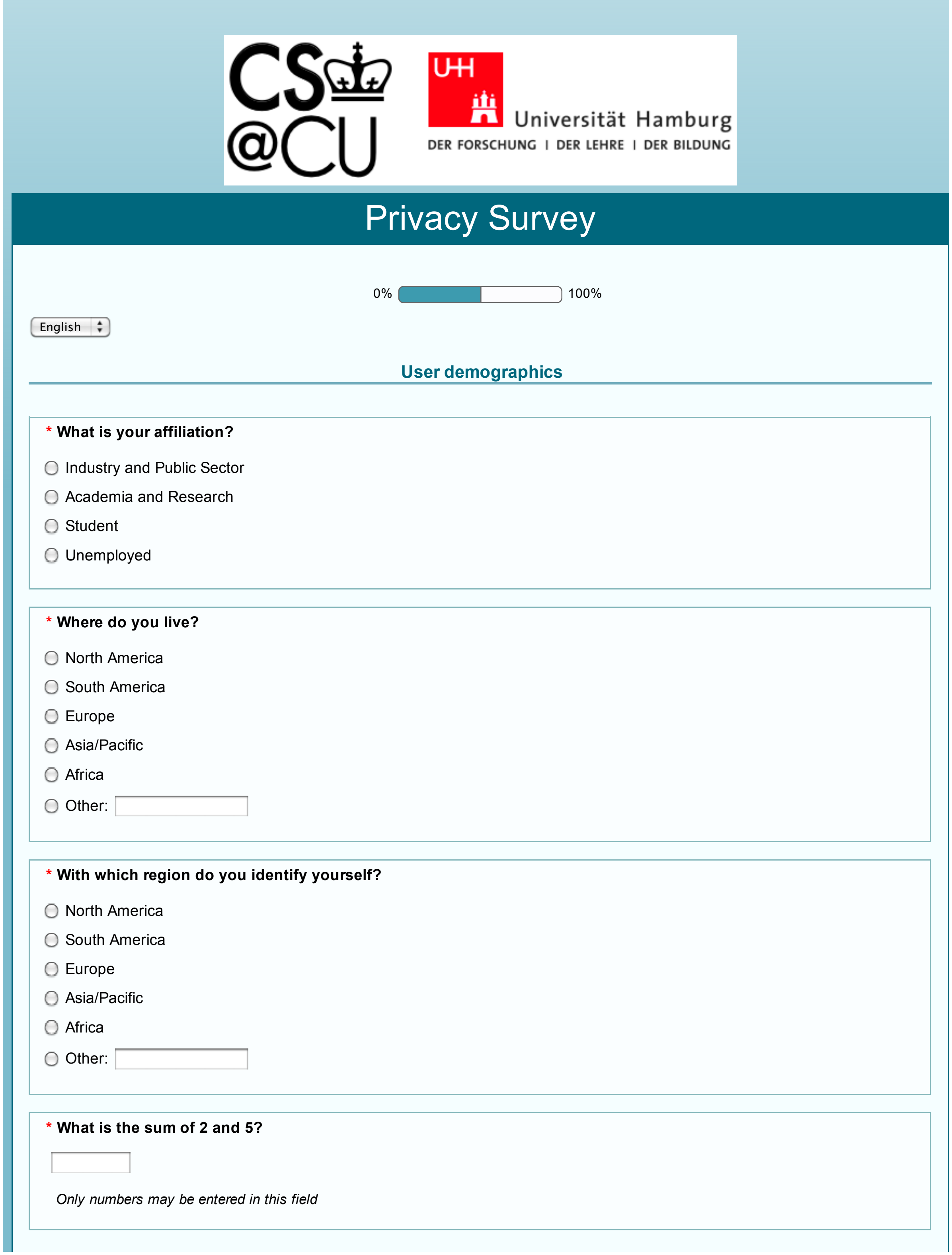
*** How critical would you rate the collection of the following data?**

(1 - very critical, 2 - critical, 3 - neutral, 4 - somewhat uncritical, 5 - uncritical)

	1	2	3	4	5
Content of documents (such as the email body)	○	○	○	○	○
Metadata (such as date)	○	○	○	○	○
Interaction (such as a mouse click to open or send an email)	○	○	○	○	○



User location	○	○	○	○	○
Name or personal data	○	○	○	○	○
User preferences	○	○	○	○	○



Privacy Survey

User demographics

* What is your affiliation?

○ Industry and Public Sector
○ Academia and Research
○ Student
○ Unemployed

* Where do you live?

○ North America
○ South America
○ Europe
○ Asia/Pacific
○ Africa
○ Other: ______

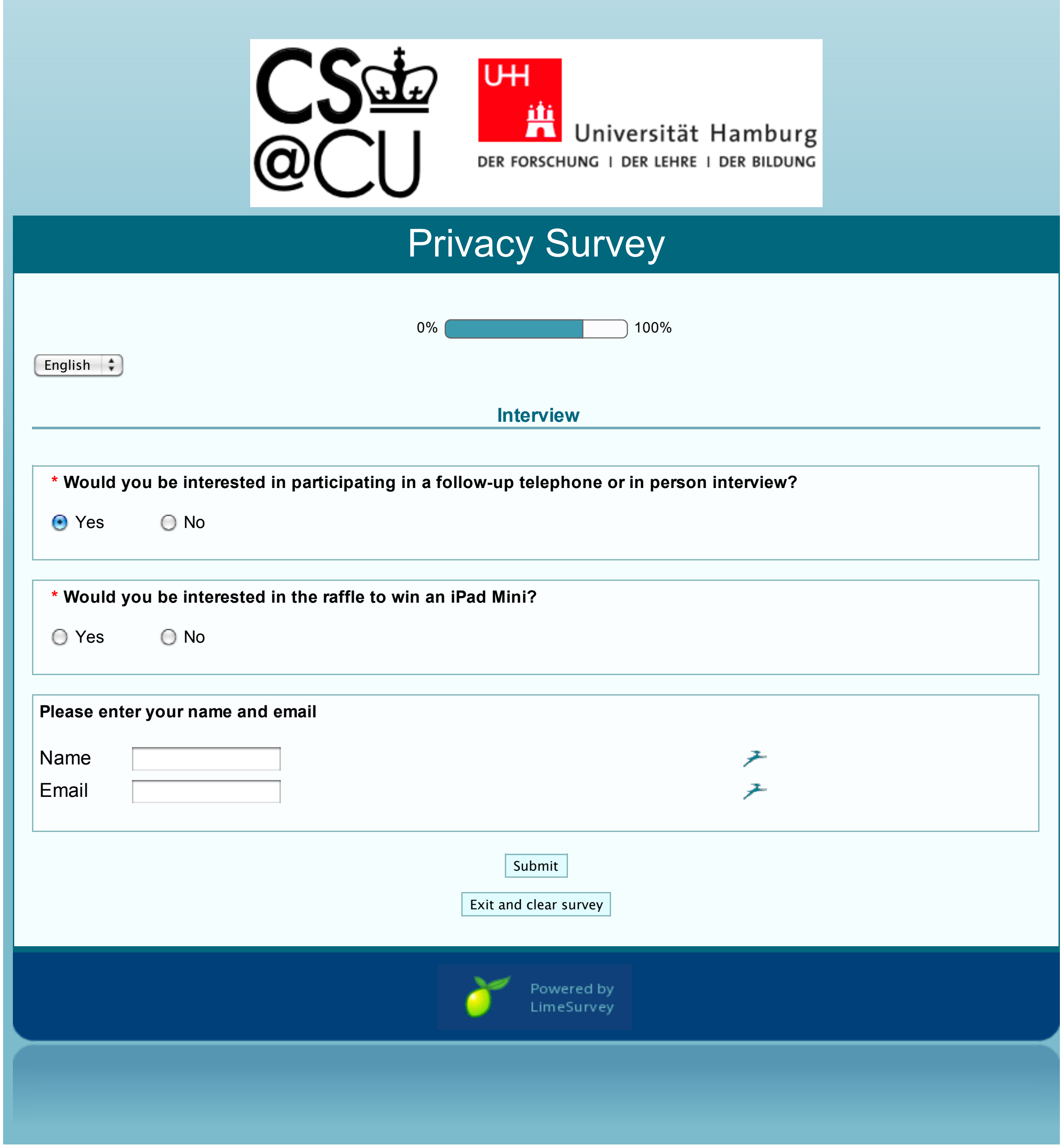
* With which region do you identify yourself?

○ North America
○ South America
○ Europe
○ Asia/Pacific
○ Africa
○ Other: ______

* What is the sum of 2 and 5?

______

Only numbers may be entered in this field



Privacy Survey

Interview

* Would you be interested in participating in a follow-up telephone or in person interview?

● Yes ○ No

* Would you be interested in the raffle to win an iPad Mini?

○ Yes ○ No

Please enter your name and email

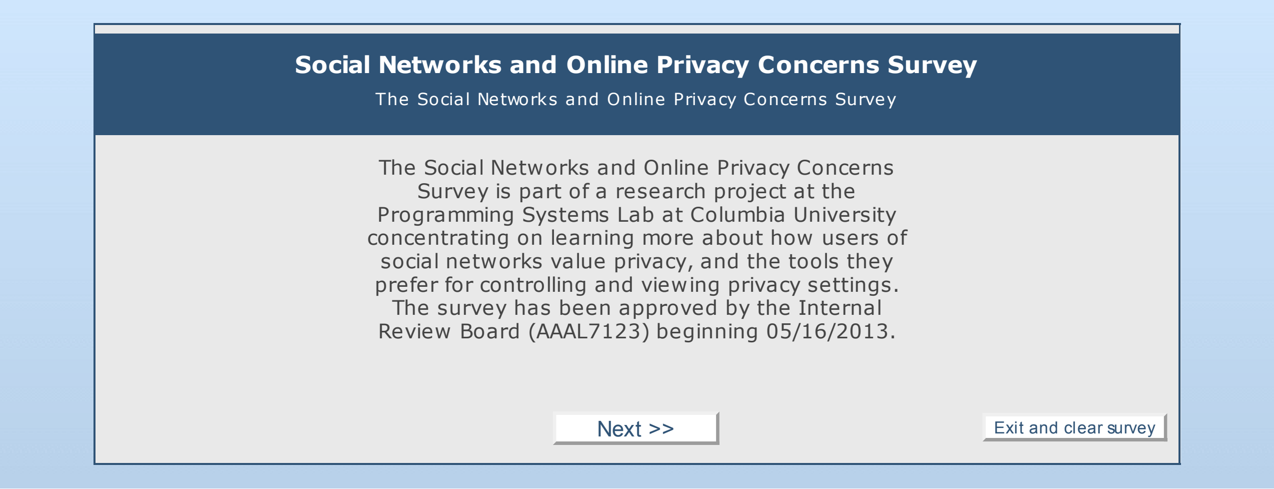
Name

Email

Appendix B

Crowdsourcing Privacy Settings User Study

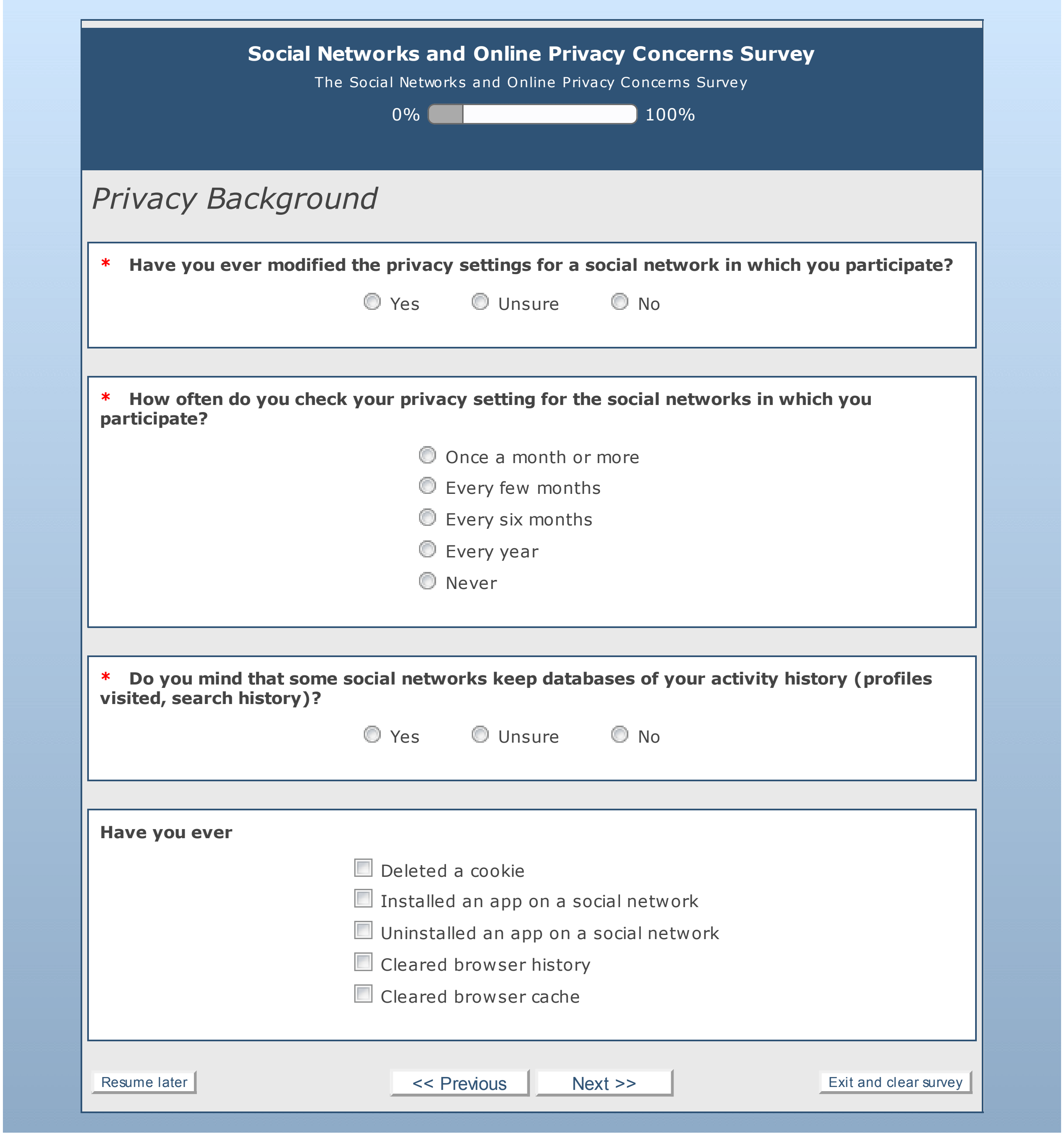
B.1 Crowdsourcing Online Survey



Social Networks and Online Privacy Concerns Survey

The Social Networks and Online Privacy Concerns Survey

The Social Networks and Online Privacy Concerns Survey is part of a research project at the Programming Systems Lab at Columbia University concentrating on learning more about how users of social networks value privacy, and the tools they prefer for controlling and viewing privacy settings. The survey has been approved by the Internal Review Board (AAAL7123) beginning 05/16/2013.



Social Networks and Online Privacy Concerns Survey

The Social Networks and Online Privacy Concerns Survey

Privacy Background

* **Have you ever modified the privacy settings for a social network in which you participate?**

○ Yes ○ Unsure ○ No

* **How often do you check your privacy setting for the social networks in which you participate?**

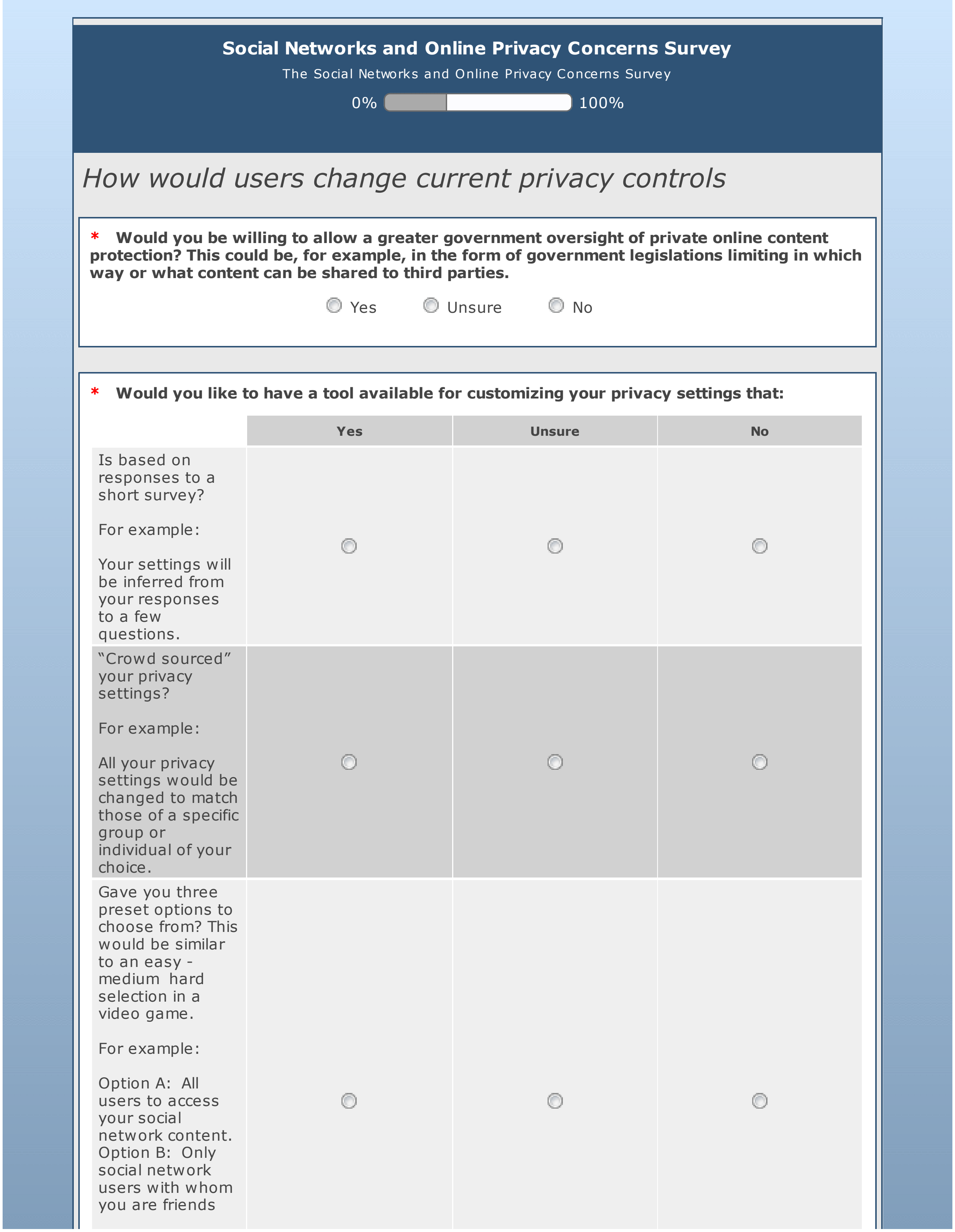
○ Once a month or more
○ Every few months
○ Every six months
○ Every year
○ Never

* **Do you mind that some social networks keep databases of your activity history (profiles visited, search history)?**

○ Yes ○ Unsure ○ No

Have you ever

□ Deleted a cookie
□ Installed an app on a social network
□ Uninstalled an app on a social network
□ Cleared browser history
□ Cleared browser cache



Social Networks and Online Privacy Concerns Survey

The Social Networks and Online Privacy Concerns Survey

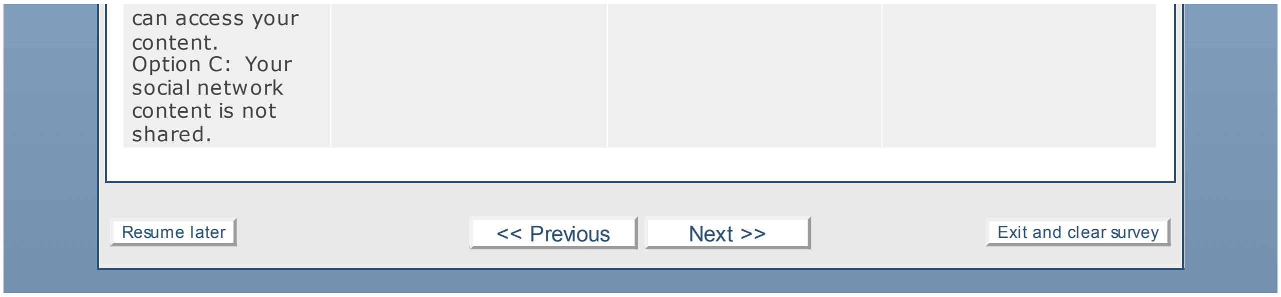
How would users change current privacy controls

* **Would you be willing to allow a greater government oversight of private online content protection? This could be, for example, in the form of government legislations limiting in which way or what content can be shared to third parties.**

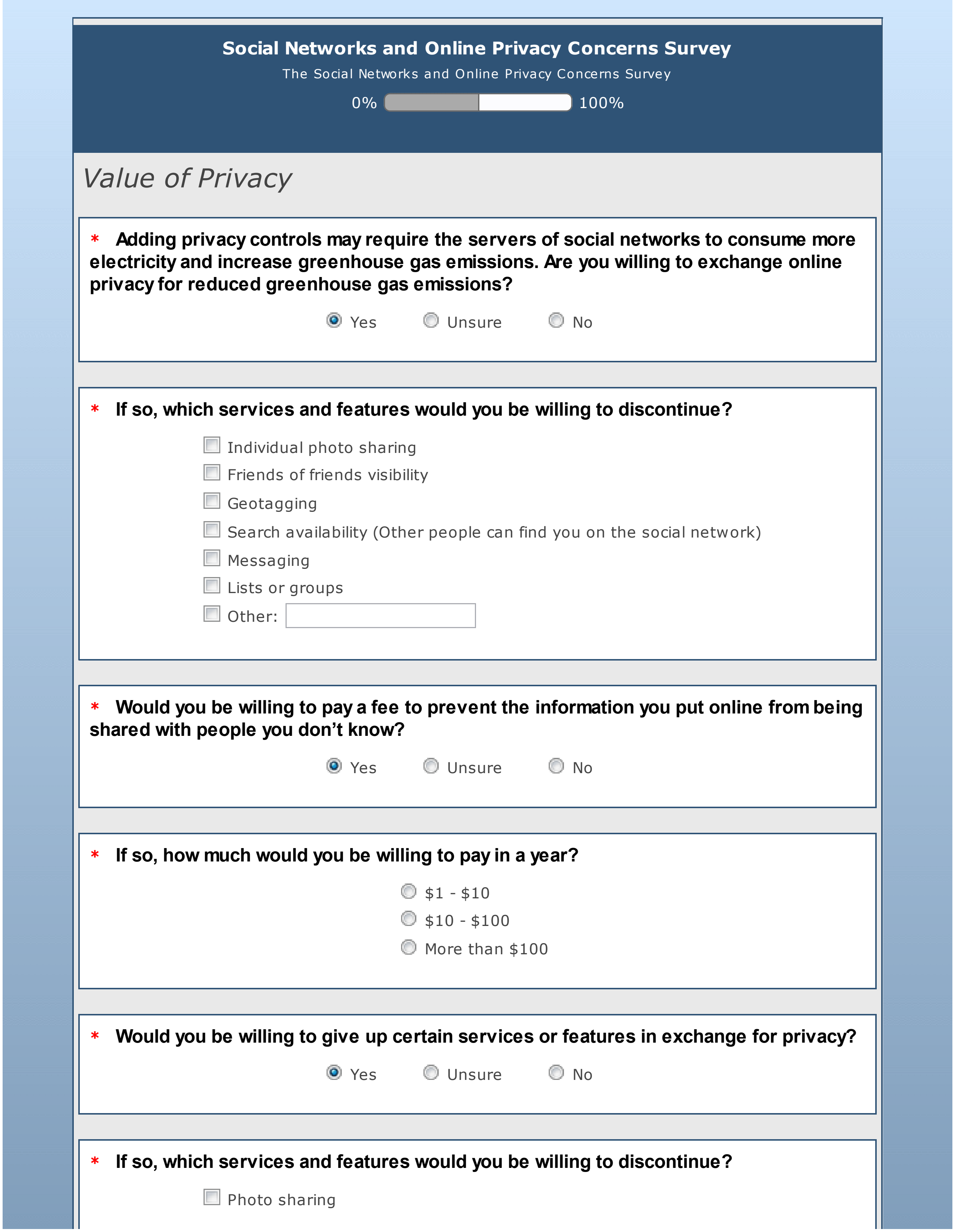
○ Yes ○ Unsure ○ No

* **Would you like to have a tool available for customizing your privacy settings that:**

	Yes	**Unsure**	**No**
Is based on responses to a short survey? For example: Your settings will be inferred from your responses to a few questions.	○	○	○
"Crowd sourced" your privacy settings? For example: All your privacy settings would be changed to match those of a specific group or individual of your choice.	○	○	○
Gave you three preset options to choose from? This would be similar to an easy - medium hard selection in a video game. For example: Option A: All users to access your social network content. Option B: Only social network users with whom you are friends	○	○	○



can access your content. Option C: Your social network content is not shared.			



Social Networks and Online Privacy Concerns Survey

The Social Networks and Online Privacy Concerns Survey

Value of Privacy

* **Adding privacy controls may require the servers of social networks to consume more electricity and increase greenhouse gas emissions. Are you willing to exchange online privacy for reduced greenhouse gas emissions?**

● Yes ○ Unsure ○ No

* **If so, which services and features would you be willing to discontinue?**

- □ Individual photo sharing
- □ Friends of friends visibility
- □ Geotagging
- □ Search availability (Other people can find you on the social network)
- □ Messaging
- □ Lists or groups
- □ Other: ____

* **Would you be willing to pay a fee to prevent the information you put online from being shared with people you don't know?**

● Yes ○ Unsure ○ No

* **If so, how much would you be willing to pay in a year?**

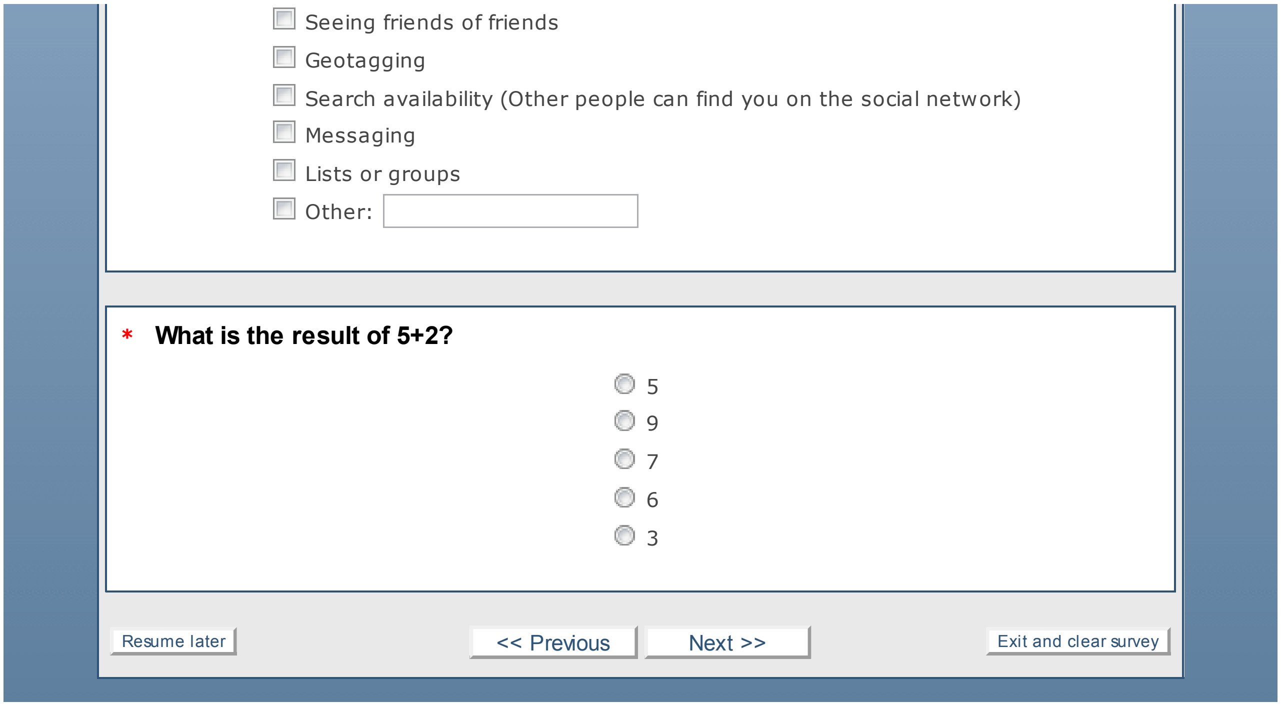
- ○ $1 - $10
- ○ $10 - $100
- ○ More than $100

* **Would you be willing to give up certain services or features in exchange for privacy?**

● Yes ○ Unsure ○ No

* **If so, which services and features would you be willing to discontinue?**

- □ Photo sharing



□ Seeing friends of friends

□ Geotagging

□ Search availability (Other people can find you on the social network)

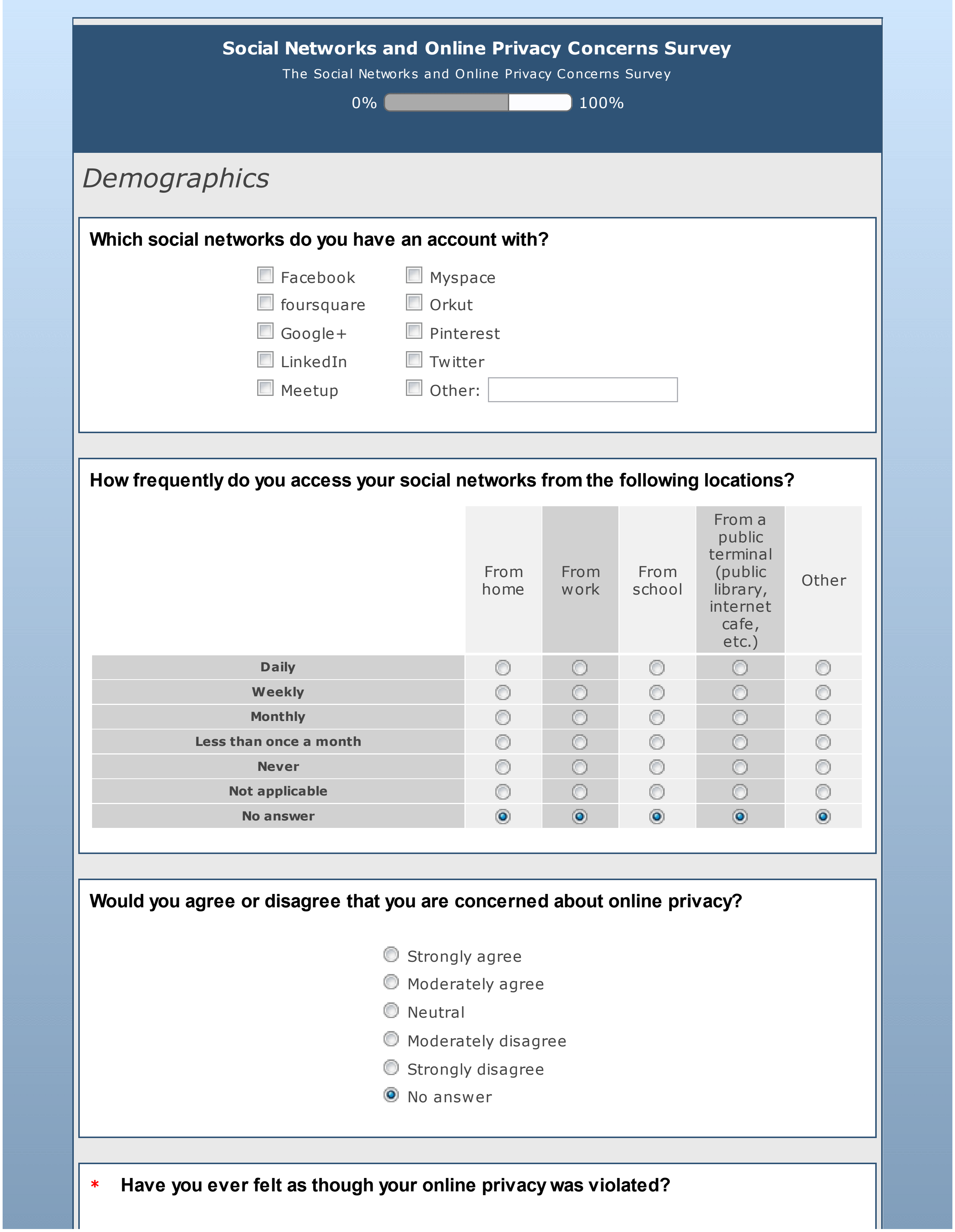
□ Messaging

□ Lists or groups

□ Other: ____

* **What is the result of 5+2?**

○ 5
○ 9
○ 7
○ 6
○ 3



Social Networks and Online Privacy Concerns Survey

The Social Networks and Online Privacy Concerns Survey

0% 100%

Demographics

Which social networks do you have an account with?

□ Facebook
□ foursquare
□ Google+
□ LinkedIn
□ Meetup
□ Myspace
□ Orkut
□ Pinterest
□ Twitter
□ Other:

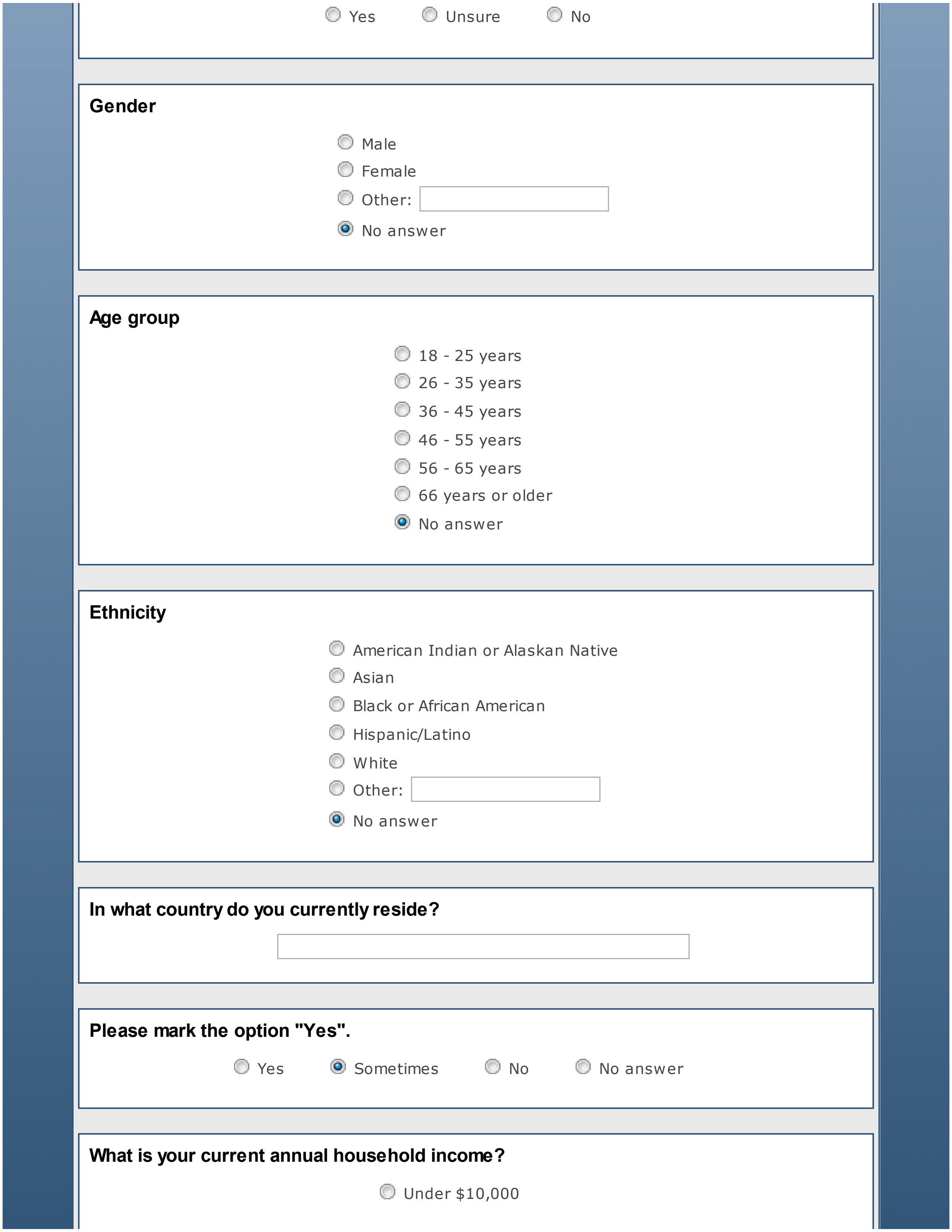
How frequently do you access your social networks from the following locations?

	From home	From work	From school	From a public terminal (public library, internet cafe, etc.)	Other
Daily	○	○	○	○	○
Weekly	○	○	○	○	○
Monthly	○	○	○	○	○
Less than once a month	○	○	○	○	○
Never	○	○	○	○	○
Not applicable	○	○	○	○	○
No answer	●	●	●	●	●

Would you agree or disagree that you are concerned about online privacy?

○ Strongly agree
○ Moderately agree
○ Neutral
○ Moderately disagree
○ Strongly disagree
● No answer

* **Have you ever felt as though your online privacy was violated?**



○ Yes ○ Unsure ○ No

Gender

○ Male
○ Female
○ Other:
● No answer

Age group

○ 18 - 25 years
○ 26 - 35 years
○ 36 - 45 years
○ 46 - 55 years
○ 56 - 65 years
○ 66 years or older
● No answer

Ethnicity

○ American Indian or Alaskan Native
○ Asian
○ Black or African American
○ Hispanic/Latino
○ White
○ Other:
● No answer

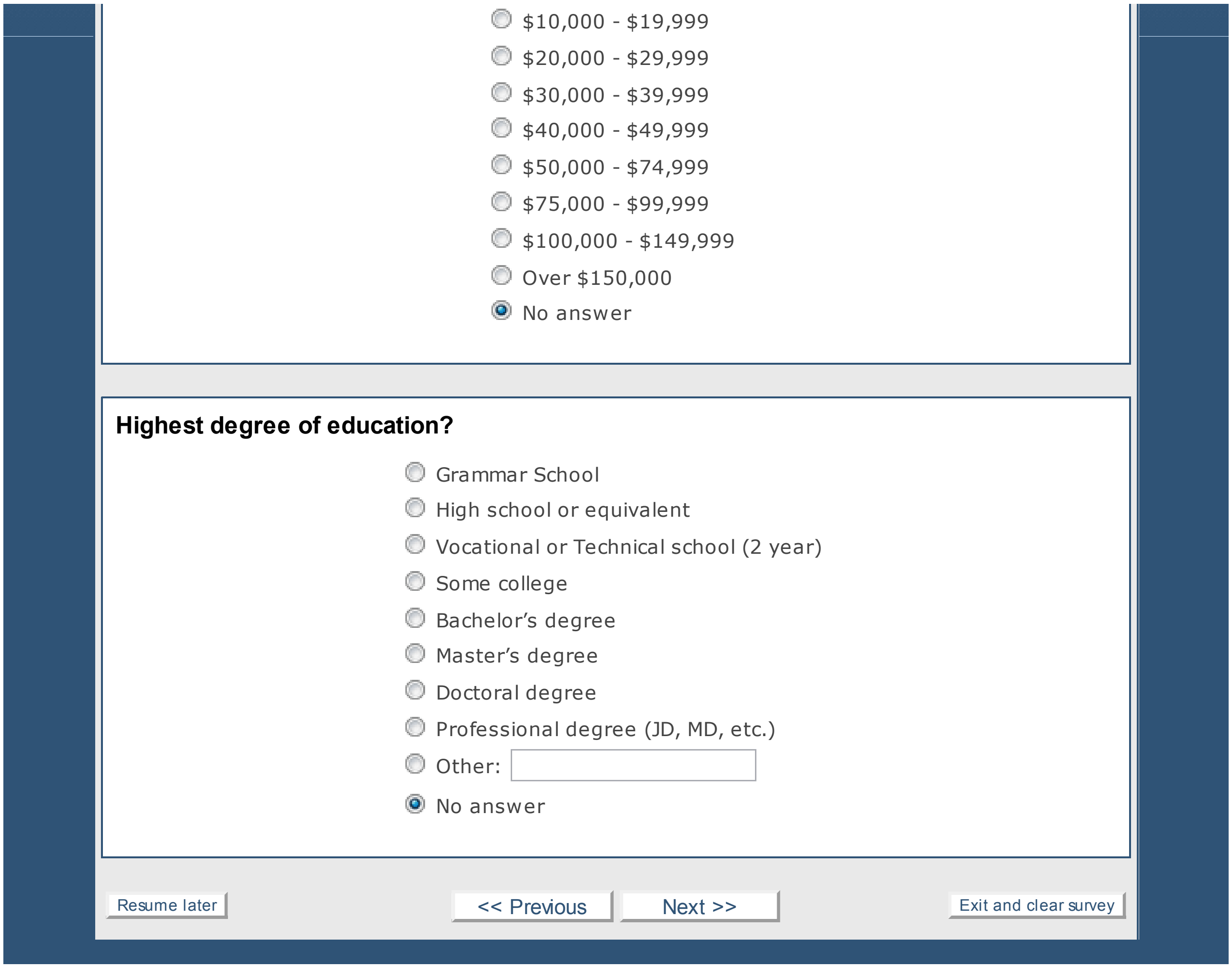
In what country do you currently reside?

Please mark the option "Yes".

○ Yes ● Sometimes ○ No ○ No answer

What is your current annual household income?

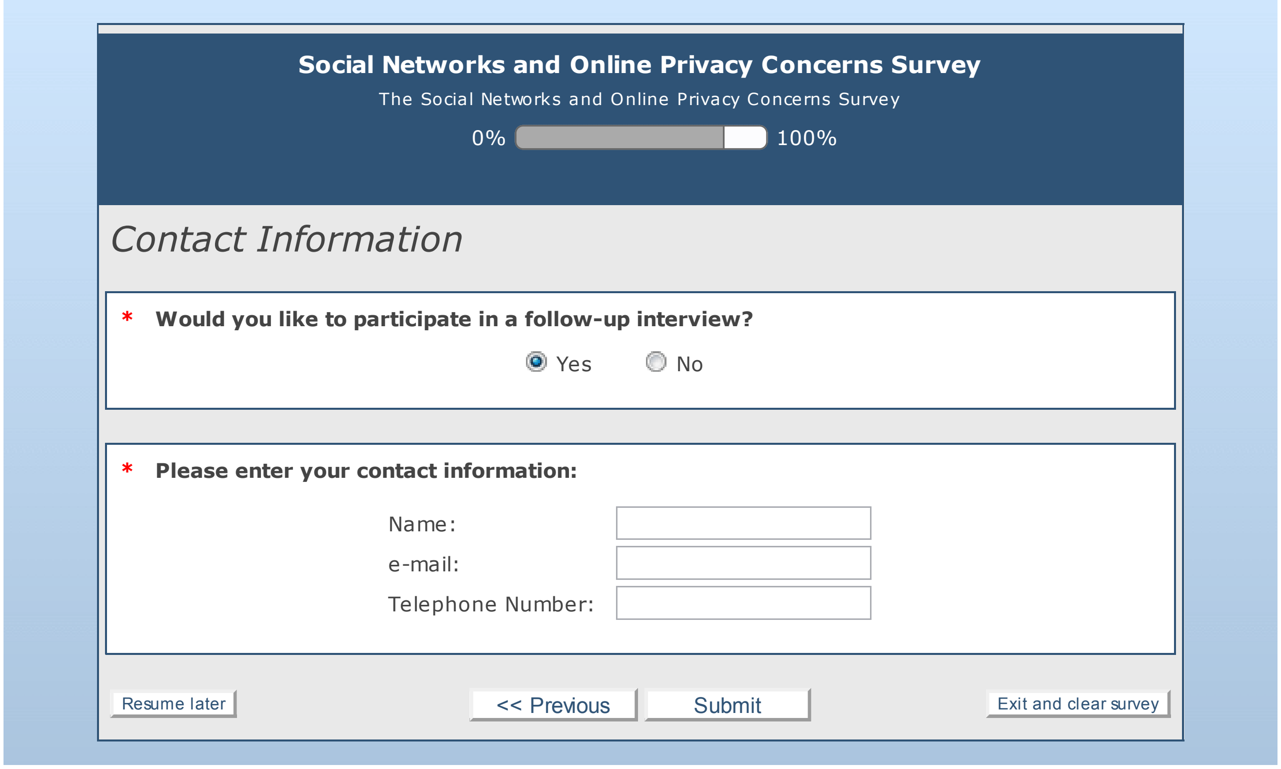
○ Under $10,000



○ $10,000 - $19,999
○ $20,000 - $29,999
○ $30,000 - $39,999
○ $40,000 - $49,999
○ $50,000 - $74,999
○ $75,000 - $99,999
○ $100,000 - $149,999
○ Over $150,000
● No answer

Highest degree of education?

○ Grammar School
○ High school or equivalent
○ Vocational or Technical school (2 year)
○ Some college
○ Bachelor’s degree
○ Master’s degree
○ Doctoral degree
○ Professional degree (JD, MD, etc.)
○ Other:
● No answer



Social Networks and Online Privacy Concerns Survey

The Social Networks and Online Privacy Concerns Survey

0% 100%

Contact Information

* Would you like to participate in a follow-up interview?

● Yes ○ No

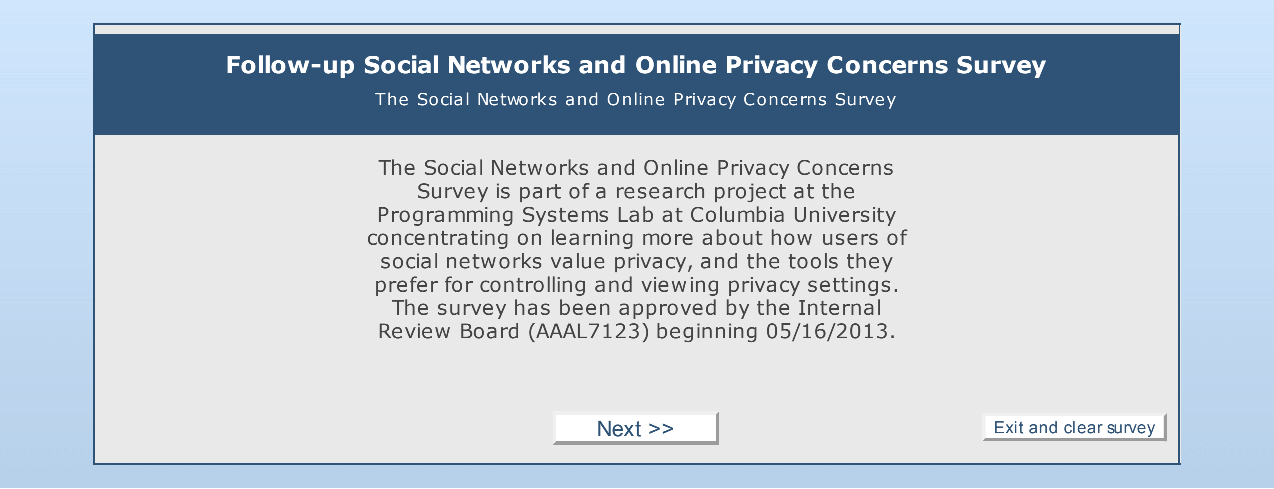
* Please enter your contact information:

Name:

e-mail:

Telephone Number:

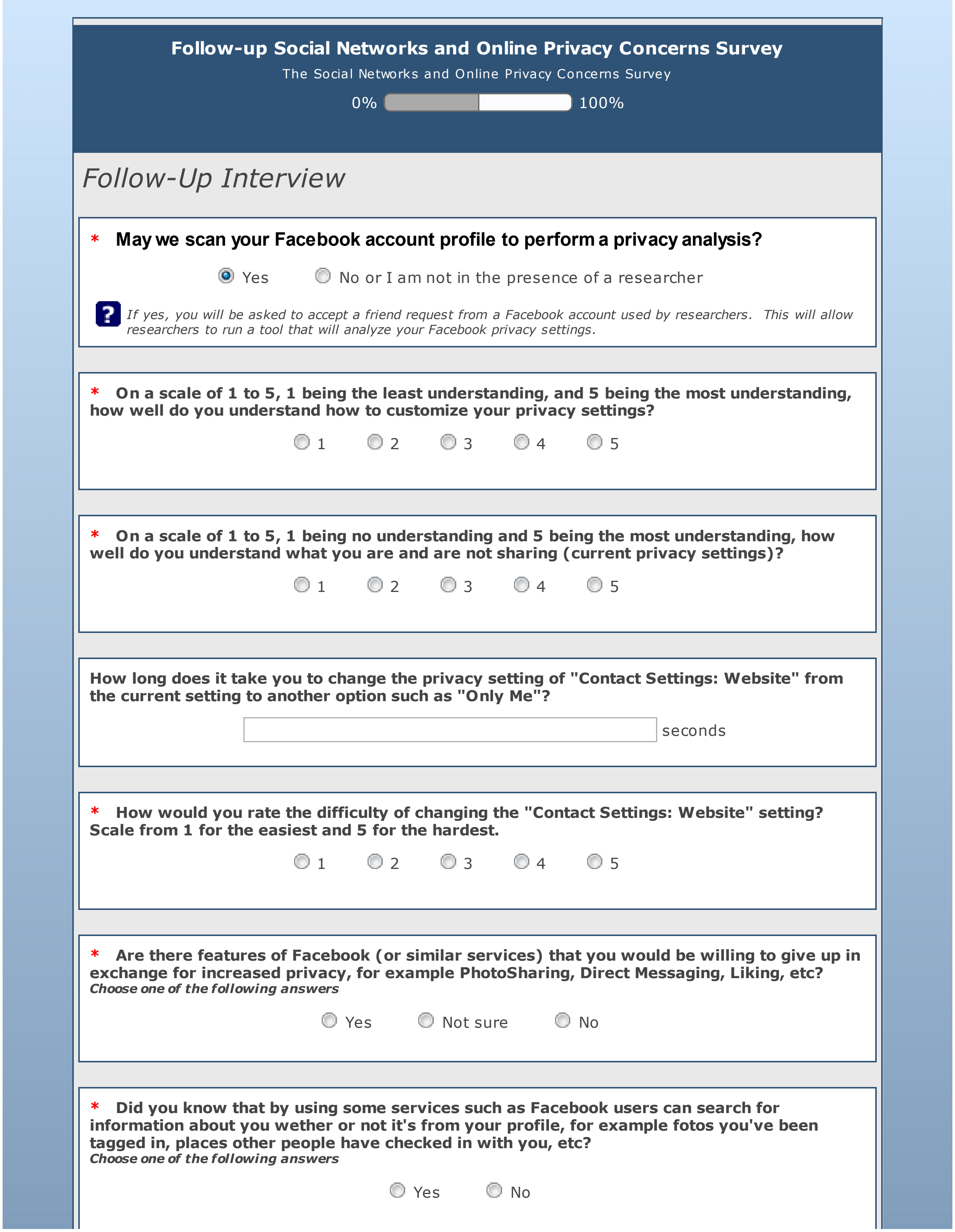
B.2 Crowdsourcing Follow-up Survey



Follow-up Social Networks and Online Privacy Concerns Survey

The Social Networks and Online Privacy Concerns Survey

The Social Networks and Online Privacy Concerns Survey is part of a research project at the Programming Systems Lab at Columbia University concentrating on learning more about how users of social networks value privacy, and the tools they prefer for controlling and viewing privacy settings. The survey has been approved by the Internal Review Board (AAAL7123) beginning 05/16/2013.



Follow-up Social Networks and Online Privacy Concerns Survey

The Social Networks and Online Privacy Concerns Survey

0% 100%

Follow-Up Interview

* **May we scan your Facebook account profile to perform a privacy analysis?**

● Yes ○ No or I am not in the presence of a researcher

? *If yes, you will be asked to accept a friend request from a Facebook account used by researchers. This will allow researchers to run a tool that will analyze your Facebook privacy settings.*

* **On a scale of 1 to 5, 1 being the least understanding, and 5 being the most understanding, how well do you understand how to customize your privacy settings?**

○ 1 ○ 2 ○ 3 ○ 4 ○ 5

* **On a scale of 1 to 5, 1 being no understanding and 5 being the most understanding, how well do you understand what you are and are not sharing (current privacy settings)?**

○ 1 ○ 2 ○ 3 ○ 4 ○ 5

How long does it take you to change the privacy setting of "Contact Settings: Website" from the current setting to another option such as "Only Me"?

__________ seconds

* **How would you rate the difficulty of changing the "Contact Settings: Website" setting? Scale from 1 for the easiest and 5 for the hardest.**

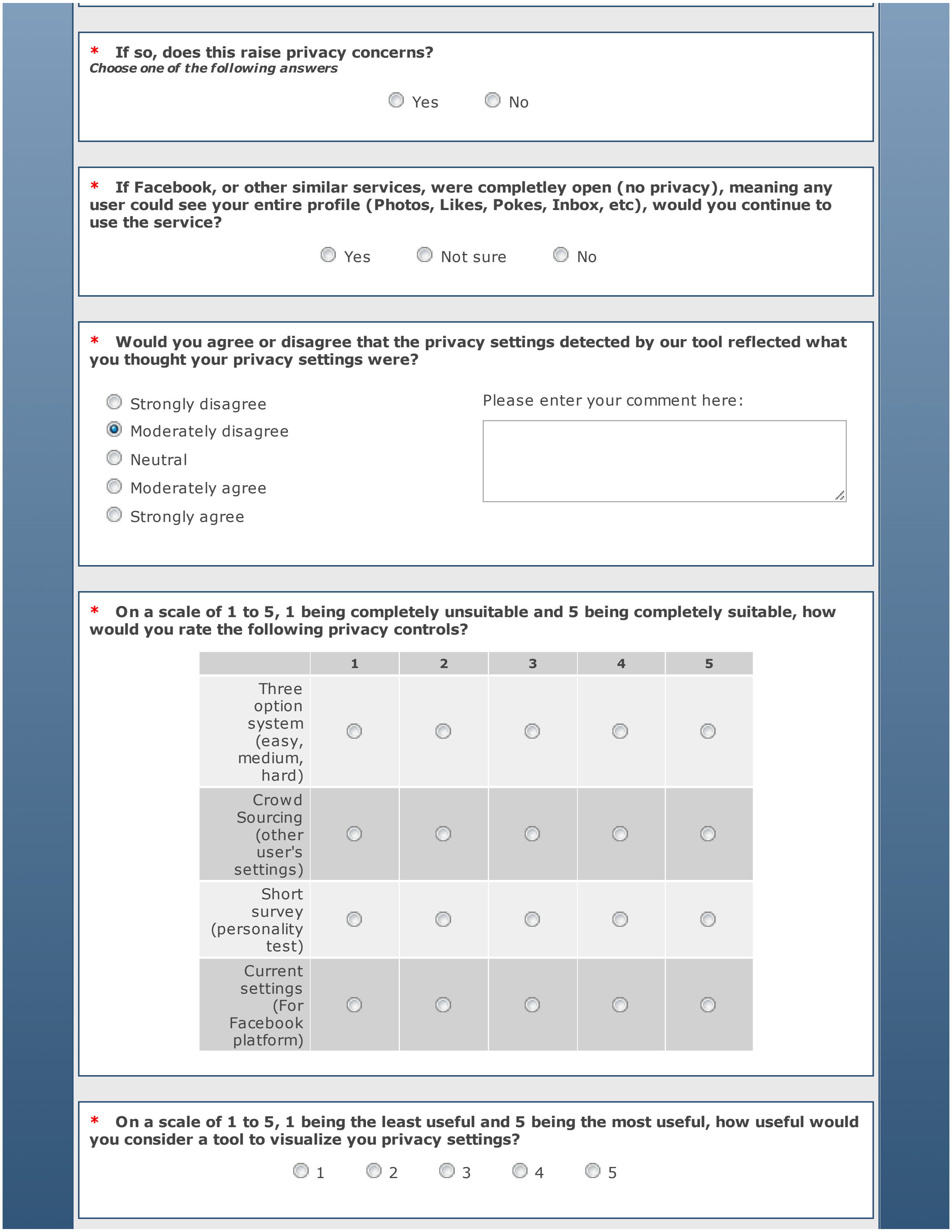
○ 1 ○ 2 ○ 3 ○ 4 ○ 5

* **Are there features of Facebook (or similar services) that you would be willing to give up in exchange for increased privacy, for example PhotoSharing, Direct Messaging, Liking, etc?**
Choose one of the following answers

○ Yes ○ Not sure ○ No

* **Did you know that by using some services such as Facebook users can search for information about you wether or not it's from your profile, for example fotos you've been tagged in, places other people have checked in with you, etc?**
Choose one of the following answers

○ Yes ○ No



* **If so, does this raise privacy concerns?**
Choose one of the following answers

○ Yes ○ No

* **If Facebook, or other similar services, were completley open (no privacy), meaning any user could see your entire profile (Photos, Likes, Pokes, Inbox, etc), would you continue to use the service?**

○ Yes ○ Not sure ○ No

* **Would you agree or disagree that the privacy settings detected by our tool reflected what you thought your privacy settings were?**

○ Strongly disagree
● Moderately disagree
○ Neutral
○ Moderately agree
○ Strongly agree

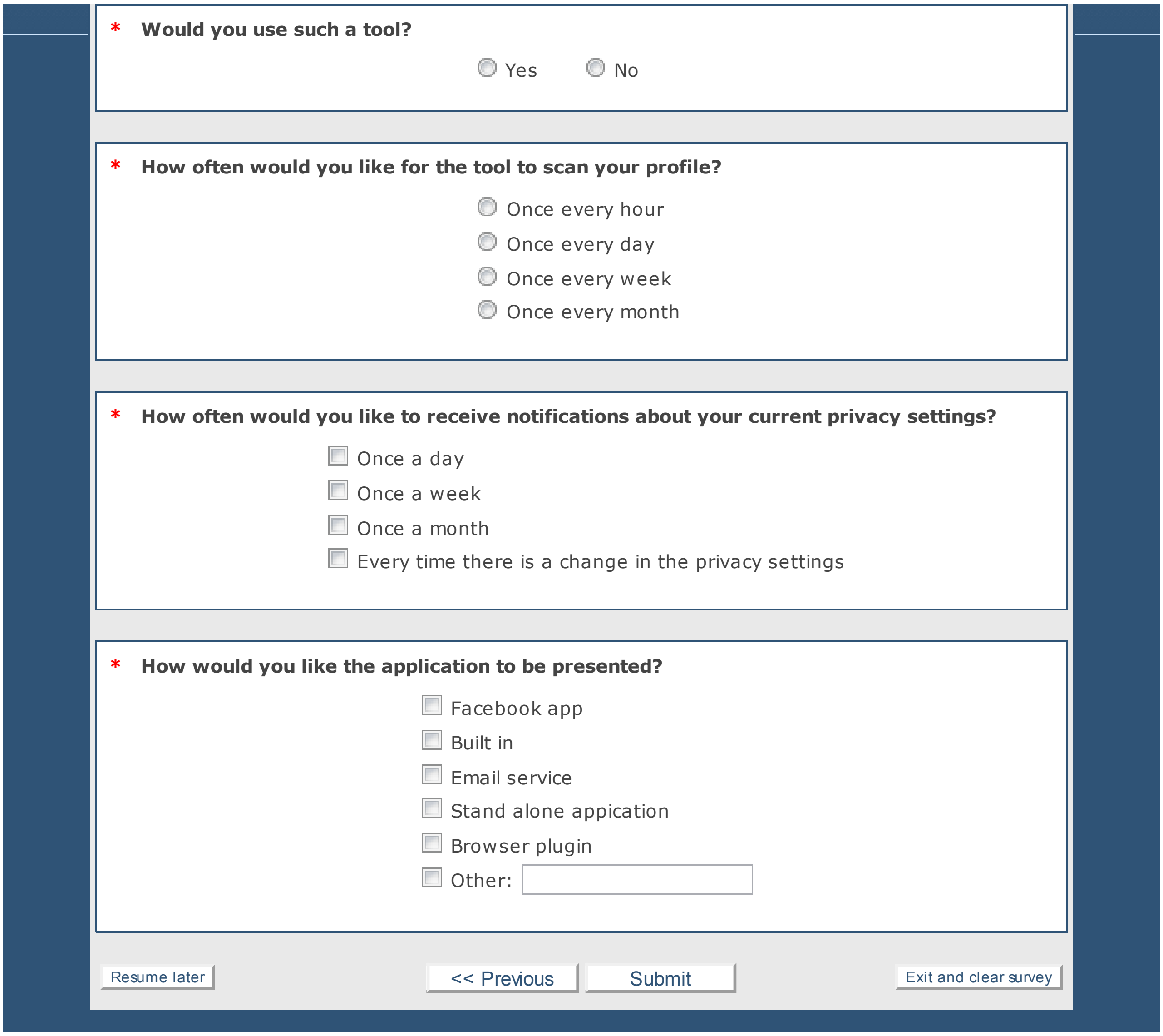
Please enter your comment here:

* **On a scale of 1 to 5, 1 being completely unsuitable and 5 being completely suitable, how would you rate the following privacy controls?**

	1	2	3	4	5
Three option system (easy, medium, hard)	○	○	○	○	○
Crowd Sourcing (other user's settings)	○	○	○	○	○
Short survey (personality test)	○	○	○	○	○
Current settings (For Facebook platform)	○	○	○	○	○

* **On a scale of 1 to 5, 1 being the least useful and 5 being the most useful, how useful would you consider a tool to visualize you privacy settings?**

○ 1 ○ 2 ○ 3 ○ 4 ○ 5



* **Would you use such a tool?**

○ Yes ○ No

* **How often would you like for the tool to scan your profile?**

○ Once every hour
○ Once every day
○ Once every week
○ Once every month

* **How often would you like to receive notifications about your current privacy settings?**

□ Once a day
□ Once a week
□ Once a month
□ Every time there is a change in the privacy settings

* **How would you like the application to be presented?**

□ Facebook app
□ Built in
□ Email service
□ Stand alone appication
□ Browser plugin
□ Other: ____________