Where to Recruit for Security Development Studies: Comparing Six Software Developer Samples

Harjot Kaur[†]
kaur@sec.uni-hannover.de

Sabrina Amft[*]
sabrina.amft@cispa.de

Daniel Votipka[¶]
dvotipka@cs.tufts.edu

Yasemin Acar[§‡]
acar@gwu.edu

Sascha Fahl[*†]
sascha.fahl@cispa.de

[†]*Leibniz University Hannover* [*]*CISPA Helmholtz Center for Information Security* [¶]*Tufts University*
[‡]*George Washington University* [§]*Max Planck Institute for Security and Privacy*

Abstract

Studying developers is an important aspect of usable security and privacy research. In particular, studying security development challenges such as the usability of security APIs, the secure use of information sources during development or the effectiveness of IDE security plugins raised interest in recent years. However, recruiting skilled participants with software development experience is particularly challenging, and it is often not clear what security researchers can expect from certain participant samples, which can make research results hard to compare and interpret. Hence, in this work, we study for the first time opportunities and challenges of different platforms to recruit participants with software development experience for security development studies. First, we identify popular recruitment platforms in 59 papers. Then, we conduct a comparative online study with 706 participants based on self-reported software development experience across six recruitment platforms. Using an online questionnaire, we investigate participants' programming and security experiences, skills and knowledge. We find that participants across all samples report rich general software development and security experience, skills, and knowledge. Based on our results, we recommend developer recruitment from Upwork for practical coding studies and Amazon MTurk along with a pre-screening survey to reduce additional noise for larger studies. Both of these, along with Freelancer, are also recommended for security studies. We conclude the paper by discussing the impact of our results on future security development studies.

1 Introduction

Human factors research is essential for improving overall computer security and privacy. In particular, developers have increasingly received research attention in the community in recent years. Previous work investigated varying research questions [33, 34, 41, 57, 87], recruiting participant samples from different populations and platforms. Participants had varying levels of experience, skills, and knowledge from the domains of software development [15, 40, 52, 57, 61], reverse engineering [23, 81], vulnerability scanning [19, 69], or software testing [82]. More so than for research on end users, recruiting expert samples is often challenging and might be time-consuming and expensive. Developers often hold well paid jobs and have high workloads, thus offering motivating incentives for their participation can be hard [14, 15, 29, 50, 68, 87]. Hence, a better understanding of the advantages and disadvantages of available recruitment platforms is valuable for the community and can help researchers to recruit the participants they need more efficiently.

In the past, researchers used various recruitment platforms including computer science (CS) students, social media, online forums, freelancers, professional developers or personal networks. For example, crowdsourcing or freelancing platforms were used to recruit developers and other experts in various fields, but required, e. g., lengthy one-by-one recruitment or extensive screening and the necessity to offer high compensations similar to their usual salaries [44, 48, 72]. On the less expensive spectrum are social media, direct networking, and snowballing, which work on a word of mouth basis, but are highly dependent on researchers' and participants' contacts [26, 52, 61]. While searching for local participants on regional forums or by using physical flyers has been done in the past, this approach suffers from the same limited reach as social media recruitment, both in number, and participant diversity [58]. Researchers also used mass email invites (e. g. for students, using university mailing lists, or developers, using GitHub and Google Play). While it is easy to reach many participants this way, these emails may be perceived as spam, and emerging privacy and data protection laws further restrain this method [14, 58].

While some previous studies discussed their recruitment experiences [29, 33, 50], to the best of our knowledge, we are the first to systematically compare participant samples with software development experience across the popular recruitment platforms used in previous work. To do so, we analyzed 59 papers studying security expert work published in the last five years and identified common recruitment platforms and experiences, skills and knowledge researchers required from their participants. While only 24 of these papers included secure development studies and recruited experienced developers, we used the entire set of papers as ground truth to build our online questionnaire. Therefore, we used surveys from previous work to develop a survey that includes questions about general and specific job experience, software development background, security experience, skills, and knowledge and demographic information. We pre-tested this survey and distributed the final version across six samples with 706 participants total. The samples we recruited include CS students, Google Play developers, members of both the Upwork and Freelancer platforms as well as Prolific and Amazon MTurk users.

The goal of our comparative online study was to answer the following research questions:

RQ1. Which general software development and specific security development experiences, skills and knowledge can researchers expect from the common recruitment platforms we identified in previous work?

RQ2. How well do samples compare, and what are the differences between them?

RQ3. What should researchers take into account when considering sampling for a security development study?

To answer these questions, we make the following contributions in the course of this paper:

Identify Common Recruitment Platforms: We analyzed previous work that included security studies with participants with software development experience, to identify common recruitment platforms. Most commonly, studies used networking or regional methods such as recruitment by flyers or at events, despite the drawbacks such as effort to recruit large numbers and regional biases. However, for the scope of this work, we decided against using them to avoid non-generalizable results that could vary between research teams. We further find a multitude of different other approaches, both online and offline. Among the platforms we used in this work, recruiting computer science students and Google Play participants was the most common in related previous research.

Survey Design: Based on survey questions used in previous work, we design, test and provide a questionnaire to collect information about programming and security experience, knowledge and skills and information about job roles and organizational structures.

Comparing Six Samples: We survey 706 participants from six samples and compare their survey responses. We find that Google Play, Freelancer, Upwork, and MTurk participants reported the most professional software development experience. Experience performing security tasks was similar across all platforms, with MTurk participants reporting the most security experience overall, and Upwork and Freelancer often performing high as well. We see especially Google Play, Upwork, and Freelancer participants reporting the most experience with specific security tasks such as authorization/authentication, input validation and using API keys. CS students and Prolific users reporting the least experience performing security tasks. Our in-depth review of each platform also identifies several differences that may be important depending on the specific study design, such as vastly different ethnic backgrounds and expertise in particular programming languages and specific areas of security.

Recruitment Advice: We use our experiences and results to give recruitment advice for future work. For example, studies that only need a small number of experienced developers to complete complex tasks should consider Upwork, which was more time consuming for recruitment, but was easier to filter for specific security skills. Conversely, MTurk is likely a better choice when a larger sample is needed as recruitment is quicker. However, MTurk participants are likely to introduce noise due to fraudulent responses; we recommend to use a pre-screening survey for improved data quality.

Replication Package: To support reproducibility of our work, we provide the following materials in a replication package [46]: the screening and final survey questions we used, formatted collection of questions we found in our literature evaluation, text used in the recruitment emails and the job posts, additional figures and tables, and the consent forms [1].

The remainder of this paper is organized as follows: In Section 2 we discuss previous works that compared and analyzed recruitment platforms in developer research. In Sections 3, 4, and 5 we present the methodology of our studies as well as their results. Following this, we address the limitations of our work in Section 6 and interpret our results in Section 7. We draw a conclusion as well as address topics for future work in Section 8.

2 Related Work

We describe and discuss related work in two key areas: Comparisons of different recruitment platforms or strategies for user studies with end-users and user studies with developers with a focus on participants with development experience.

Comparing Recruitment Platforms and Strategies: The impact of recruitment platforms has been widely discussed for participant recruitment. In 2017, Peer et al. empirically evaluated CrowdFlower and Prolific Academic as alternatives for MTurk. They report higher participant naivety, honesty and diversity on CrowdFlower and Prolific as compared to MTurk, but the data quality differed. Prolific performed similar to MTurk, but CrowdFlower offered the lowest data quality [62]. A 2017 case study by Bentley et al. compared MTurk and SurveyMonkey to a sample recruited by a market research

[1]You will also find the paper's replication package in its accompanying website [1].

company in the context of product design. They found that most results fall into a 10% margin of error, which fits their sample sizes of 150 per survey platform [24]. In 2014, Schnorf et al. compared six survey sample providers and panels (e. g., Google Consumer Surveys). They distributed a survey regarding end-user privacy comfort to investigate potential differences between the survey providers by recruiting between 835 and, 1115 participants per sample. They found that the results differed based on platforms because some recruit random users while others use probabilistic methods or opt-ins [67]. In 2019, Redmiles et al. compared Amazon MTurk with a census-representative sample from a web panel and probabilistic sample via telephone. They found that the results from MTurk were closer to the probabilistic sample and therefore the general US public than the web panel, suggesting that the choice of platform has an impact on results [64]. A 2019 work by Chandler et al. compared MTurk and Prime Panels (an online research panel aggregate). They find similar data quality but more diverse participants on Prime Panels than on MTurk with more direct access to lesser represented user groups, albeit with inevitable trade-offs [28]. While the above studies focused on the recruitment of end-users, our work is the first to compare six samples of expert users with a focus on participants' software development experience.

Studies with Security Information Workers: Since misunderstandings and over-estimation of capability could be a prominent issue while recruiting participants with self-assessed coding skills, Danilova et al. developed and evaluated screening questions for efficacy by recruiting several batches of participants with and without programming skill from multiple platforms. They found that 42% of Clickworker users advertising programming skill did not meet their criteria and could be filtered out [30]. This study is tangential to ours, as they focused on identifying effective questions for developer screening, while our focus is on recruitment strategies using questions from the existing literature. A 2019 literature review by Tahaei et al. addressed the status quo of developer-centred security research, summarizing 49 papers with developer studies that were published in 19 conference proceedings or journals between 2005 and 2019 [73]. We build our literature review on their work, however, we focus on recruitment strategies. Furthermore, in a 2021 literature review, Kaur et al. analyzed and found 557 relevant publications including regular users and experts in security research. Of these, they analyzed 48 expert studies in depth. Among other goals, they investigated participant demographics, how they were recruited and ways to align research objectives with research methods [47]. In contrast, for the current work, in addition to the literature review of previous security research to identify common developer recruitment methods, we survey developers from six popular platforms identified in our literature review and compare both general and security demographics. To the best of our knowledge, the recruitment strategies for developer studies have not yet been comprehensively researched. Various studies recruit computer science students as a representative for developers due to their technical background and because they are easier to recruit. If developers were recruited, the criteria for deciding who is an expert are different for each research study. Our research differs from previous works by providing both a comprehensive literature review of top tier developer studies, and a larger-scale comparison of different recruitment platforms, all of which were used in previous research in this domain.

3 Security Studies with Software Developers

In the following section, we investigate recruitment strategies and survey questions in security studies with experienced software developers. We aim to gain insights into common recruitment strategies and the experiences, skills and knowledge previous studies required from their participants. Therefore, we collected and reviewed five years of relevant research published at important security, privacy and HCI venues. We did not aim for an exhaustive literature review across all potential venues from the beginning of studying developers. Instead, our goal was to learn recent common practices. The literature review is the foundation of the comparative studies we conducted in Section 4. Figure 1 gives an overview of our overall methodology.

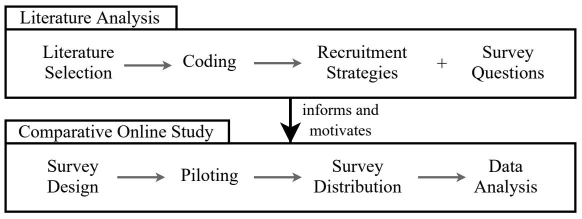


Figure 1: Methodology overview: The literature review informed and motivated our comparative online studies.

3.1 Literature Selection

We broadly selected publications in the field of usable security and privacy that conducted user studies with security experts. Although we only survey developers, studies with different types of experts are often related and similar in nature, and therefore including them gives us a wider net, which helped us design a fitting survey questionnaire. We focused on works published between 2016 and 2020 at the top (usable) security and privacy, and human computer interaction venues: the USENIX Security Symposium (SEC) [13], the ACM Conference on Computer and Communications Security (CCS) [2], the IEEE Symposium on Security and Privacy (S&P) [7], the Network and Distributed System Security Symposium (NDSS) [8], the Symposium on Usable Privacy and Security (SOUPS) [11], the Human Factors in Computing

Systems (CHI) [6], as well as the workshops Usable Security (USEC) [8] and its European counterpart (EuroUSEC) [8]. These venues were chosen based on two conference lists for computer security and HCI, filtered by citation counts [3, 12]. Besides selecting the most relevant ones, some were further down and were chosen due to their focus on usable security and HCI.

For all included venues except CHI, two authors independently reviewed all proceedings published between January 2016 and September 2020 by reading the abstracts and titles. For CHI, where the publication volume is significantly higher than at the other venues, we used a different approach. Instead, we searched the CHI proceedings for keywords related to our context (*secur*, *priva*, *develop*, *admin*, *expert* and *program*), manually investigating the results. Additionally, we compared our set of papers with the developer papers listed in two other publications [47,73]. This brought up three additional papers, but we neither learned novel recruitment strategies nor survey questions and therefore did not add them to our list. Overall, we initially found 95 papers that at least one author identified as relevant. We reviewed all conflicts and removed 21 papers that, after closer inspection, e. g., did not include a developer study, leaving us with 74 papers for further analysis.

3.2 Literature Survey

For the remaining set of papers, we collected information about participant recruitment and survey questions if available. We extracted the following information: type of participants (e. g. software developers or system administrators), recruitment platform (e. g. MTurk or CS students), number of participants, and compensation amount and type. Two authors independently reviewed each paper for the above information in detail. Disagreements were immediately discussed and resolved. In the course of the detailed analysis, we removed 15 more papers for either a missing security focus or not recruiting expert users, leaving us with 59 papers in the final set (cf. Table 3 and 4 in Appendix A).

For papers that included studies with both end-users and expert users, we only considered expert user information. To collect all information from each paper, we first checked if it was included and available as part of the paper, the appendix, or a replication package. For questions, we decided to focus purely on surveys and excluded questions from interview studies. Interviewers often use open-ended questions and encourage participants to elaborate their answers, which works well for explorative interviews, but might not translate well to quantitative survey questionnaires.

Overall, we found participant recruitment information in 58 papers and 363 questions in 45 papers. In ten cases we could not find the information in the paper and contacted the authors. All but one acknowledged our request and provided us with the necessary information. One research team did not retain the original survey questions and could only provide us with rough estimates of the used questions. We assigned all extracted survey questions and answer options to one of the following categories:

General. Demographics such as age, gender, or education.
Experiences, Skills, and Knowledge. Security and programming experiences, skills and knowledge.
Scales. Established scales such as the System Usability Score (SUS) or the Secure Software Development Self-Efficacy Score (SSD-SES).
Specific. Specific questions for studies, e. g. self-assessment of task success or failure.

We excluded specific questions from our survey if they were narrowly focused on e. g. a specific tool or area of development. In case multiple papers included questions with identical or similar phrasing, we merged them while keeping track of their origin.

3.3 Results

Below, we present and discuss the results of our literature survey used to inform and motivate the comparative studies in Section 4. However, we do not intend to systematize previous work (cf. [73] and [47] for respective systematizations).

24 of the analyzed papers recruited participants with software development experience, ten recruited system administrators or operators, and another 24 recruited security experts. While these security experts all had computer science and security backgrounds, they differed from each other, e. g. in terms of experience in a certain job role, skills, or security certifications. In seven papers, we could not find sufficient details to determine the participants' job role. One paper did not provide exact participant numbers. 23 papers offered fixed payments of varying amounts, some using gift cards. One paper used performance-based payments, and six raffled a prize among all participants. For the other papers, 21 did not mention any reward, and 11 stated that they did not reimburse participants. Overall, we identified 25 recruitment platforms. In 12 papers, recruitment was described at a superficial level, such as e. g. *social media ads* [49] or *online cold calling* [53]. We assigned all 25 strategies to six categories (number of papers in parentheses):

Unsolicited Emailing (10): Papers in these categories sent unsolicited emails by collecting participants' contact information. Example email collection platforms are GitHub (5) and Google Play (5).
Social or Regional Contacts (75): Strategies based on some form of professional or personal network (29), snowball sampling (10), as well as recruiting through security related events (13) or regional expert meetups (4). Also includes the distribution of flyers (6), Craigslist (2), and the recruitment of computer science students (11).
Social Media (10): Posting study information on social media platforms such as Twitter (5), Facebook Groups (1) and

Ads (2), and the chat software Slack (2).

Online Forums/Blogs (33): Strategies relying on discussion platforms such as Reddit (8), online forums (8), mailing lists (15), and blogs (2) dedicated to computer science topics.

Paid Workers (4): Freelancing or crowdsourcing platforms such as Prolific (1), Upwork (1), and Freelancer (2).

Networking (8): Professional networking platforms such as LinkedIn (7) or its German counterpart Xing (1).

4 Comparative Study of Recruitment Platforms for Security Developer Studies

Based on the results in Section 3 we designed, pre-tested and conducted a comparative online survey study with six samples of developers. We collected their demographic information, as well as data about participants' security and programming knowledge, skills and experience with an online questionnaire.

Before we recruited participants and conducted the surveys, we pre-tested our survey with cognitive interviews [63] with members of our research group who were not part of this project. This allowed us to gather insights into how participants interpret and answer questions. During our cognitive interviews, participants shared their thoughts as they answered each survey question. We used our findings to iteratively revise and adapt the survey questions and answer options to minimize bias and maximize validity.

In a second pre-test, we refined our survey with two rounds of pilot studies on Prolific with 20 users each, similar to approaches in other works [79]. To screen, we used Prolific filters based on self-reported computer programming and software development experience. We slightly adjusted and improved our phrasings for the final survey based on the results of our pilots.

For recruitment purposes, we created a short screening survey (cf. replication package [46]), that inquired about software development experience, current job role and gender to be able to filter eligible participants wherever necessary. For our final survey, we exclusively invited participants who claimed to have experience as a software developer in our screening survey. As we could not confirm the self-reported development experience of Prolific and MTurk users, we used two additional programming questions from Danilova et al. [30] in a new screener and repeated these samples. Finally, we conducted a priori power analysis to determine the number of required participants for our statistical tests. We used standard assumed effect sizes gathered from literature (0.25 for Kruskal-Wallis, a medium 0.3 effect size for Chi-square, and 0.8 for Mann-Whitney-U). All analysis suggested that at most 325 overall and 29 participants per group would be required, which we exceeded by far in our recruitment.

4.1 Survey Structure

We built the survey based on questions we extracted from papers in Table 3 and the 2020 StackOverflow Developer Survey [10]. We copied questions and response options directly whenever possible to ensure comparability. However, we added answer options or modified question phrasings when appropriate (e. g. we modified the gender answers to be more inclusive, following current best practices [5]).

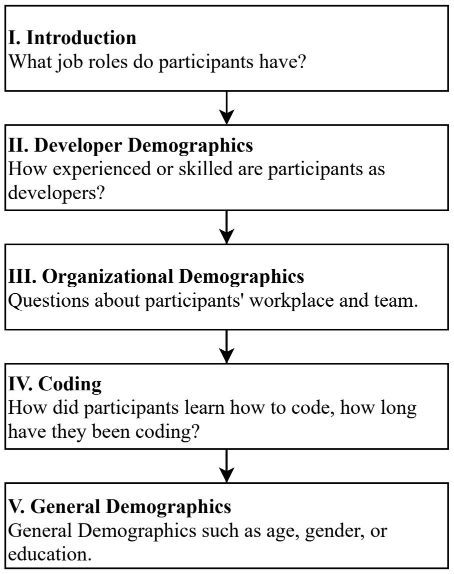

I. Introduction
What job roles do participants have?
II. Developer Demographics
How experienced or skilled are participants as developers?
III. Organizational Demographics
Questions about participants' workplace and team.
IV. Coding
How did participants learn how to code, how long have they been coding?
V. General Demographics
General Demographics such as age, gender, or education.

Figure 2: Illustration of the survey flow.

The survey structure is outlined in Figure 2. Overall, the survey consisted of five sections with 46 questions. The sections ranged from general to specific job experience, to organizational, to coding and finally demographic questions. The complete survey is listed in the replication package [46]. We distributed the survey in English, but translated the survey for German students.

I. Introduction. In the first section, we asked participants about their experiences in different job roles (cf. `Q1 – Q2`).

II. Software Development Background. We asked participants specific questions about their experience as software developers, such as how they learned about programming, how proficient they were in prominent programming areas, how much experience they had in these areas and if it included the integration or deployment of security mechanisms (cf. `Q3 – Q12`).

III. Organizational. This section contained questions addressing participants' workplace, including number of employees at their organization, size of their team, and if their work included security-relevant tasks and decisions (cf. `Q13 – Q19`).

IV. Programming. This section had questions concerning programming experience, including the expertise with different programming languages, certifications, as well as participation in security events, Capture The Flag (CTF) contests and bug bounty programs. We also asked the vulnerability identification and mitigation questions from the secure software-development self-efficacy scale (SSD-SES) [2] (cf. `Q20 - Q32`).

V. General Demographics. Finally, we asked about general demographics, e. g., gender, age, ethnicity, education, and employment status (cf. `Q33 - Q46`).

4.2 Survey Distribution

We distributed our survey on six different recruitment platforms we identified in Section 3: MTurk, Prolific, Upwork, Freelancer, Google Play and computer science students. Table 1 summarizes the recruitment platforms we used and illustrates their unique deployment characteristics, including options to filter participants. Although we are not aware of previous work that recruited participants with software development experience from Amazon MTurk, we chose to include it because of its general popularity in usable security and privacy research and to investigate to what extent it can be used in future security developer studies.

For MTurk participants, we required an approval rate of at least 95% and 100 or more jobs completed. On Prolific, we filtered for participants who had self-reported computer programming or software development experience. As we were unable to confirm the self-reported programming knowledge of Prolific or MTurk participants, we additionally added two programming skill questions from Danilova et al. [30] designed to test development skills (cf. replication package [46]) to these screening surveys and redid both platforms. On Upwork, we focused the recruitment on freelancers who had at least entry-level experience, a job success rate of 90% and were conversational in English. For participants on Freelancer.com, we used the built-in search and filtered for users based on the term 'Software Development'. On both freelancing platforms, we used user profiles to confirm participants' experience. We extracted public contact email addresses from Google Play Store applications and contacted 76,978 developers. To recruit computer science students, we contacted ten U.S. and five German universities that offered a CS degree program and kindly asked colleagues to distribute the survey between their students. We assumed that both CS students and Google Play developers have programming skills on at least a beginners level due to their professions.

Excluded Platforms: We did not recruit participants from all platforms we identified in previous work (cf. Table 1). Although previous studies used GitHub to recruit participants [14, 16, 85], we decided against it, as extracting emails from GitHub commit messages violates their terms of service and might also constitute a violation of the European GDPR. We also excluded platforms that vary significantly between research teams, making it unlikely to produce generalizable results. This includes platforms that depend on the authors' contacts, such as Twitter or other social networks. Similarly, we excluded platforms requiring exclusive access to small or local groups, such as specialized companies or developer meetups. Additionally, our own experience and feedback from some of the authors of papers in our review who used, e. g., social media or security events for recruitment (Table 3) illustrates that recruiting from these platforms usually resulted in smaller samples and higher effort (e. g., contact 20 mailing lists for 15 participants).

[2]We excluded the SSD-SES questions regarding security communication since we were mainly interested in skills and knowledge and aimed to keep the survey short.

4.3 Data Analysis and Quality

Our results only include quantitative data points. We use Kruskal-Wallis (KW) as a non-parametric equivalent to the one-way ANOVA to compare multiple independent groups and for ranked categories (e. g. Likert scales). For unranked categorical questions, we use the Chi-square test (χ^2). In case of the SSD-SES, we used a more appropriate Mann-Whitney U test. Procedurally, we first performed omnibus tests, which were then followed by pairwise comparisons. We assume an alpha level of $\alpha = .05$ for significance in hypothesis tests and corrected our results using the Benjamini-Hochberg procedure.

To improve data quality, we removed invalid participants, including 68 participants who reported a contradicting lack of experience as a developer in the final survey despite reporting otherwise in the screening survey. We further excluded 168 participants who did not finish the survey and five who gave identical answers or wrote nonsensical comments in free-text responses. We checked completion times, but did not find anyone who finished the entire survey in less than three minutes (we used estimated completion times identical to the Stack Overflow 2020 developer survey [10]). In total, we excluded 241 participants, leaving us with 706 valid responses.

4.4 Ethics

None of the involved institutions required a formal IRB approval. However, we only used previously established questions that always included options to decline to answer. Furthermore, every participant agreed to our consent form with detailed information about the study, responsible researchers and contact information, risks, benefits as well as privacy and participant rights. At the end of the survey, participants had the chance to not submit their answers to exclude them from our analysis. Our survey did not collect any PII except for the email addresses of participants interested in the raffle, which were deleted after the raffle was done. We stored the collected data on our encrypted cloud server, which only involved au-

	MTurk	Prolific	Upwork	Freelancer	Google Play	CS Students
Recruitment:						
Type	Job post	Job post	Job post	Job post	Email invite	Email invite
Duration	Dec'21-Jan'22	Dec'21	Jan'21-Feb'21	Mar'21	Feb'21-Mar'21	Feb'21-Mar'21
Participants:						
Started survey	111	126	112	111	168	316
Finished survey	103	122	107	107	107	232
Valid (total)	101	122	72	100	103	208
Screening:	Yes	Yes	Yes	No	No	No
Compensation:						
Screening	\$0.52	\$0.52	-	-	-	-
Final survey	\$5	\$5	\$5	\$5	Raffle (\$50 gift cards)	Raffle (\$50 gift cards)

Table 1: Survey distribution details for all six recruitment platforms we used.

thors could access. Additionally, we used random six-digit numbers to identify valid submissions for compensation, but they were not stored and processed further in any other way.

Compensation depended on the platform, but we aimed to award at least US federal minimum wage. For the screening surveys on Prolific and MTurk we paid \$0.15 for one minute of work which was increased to \$0.52 for three minutes in the rerun, while we awarded \$5 for the full survey. Although we do not have data on the exact survey completion times, Prolific reported rates well above the US federal minimum wage. Participants we needed to contact via email had the chance to take part in a raffle for 20 \$50 gift cards.

5 Results

In the following section, we report and discuss the results for our survey across all six samples. As we could not find significant differences between them, we merged the English and German student samples and refer to them as *students* collectively to mitigate the regional bias when examining samples from different countries. Since some of our questions allowed multiple response selections, the percentages we report may not always add up to 100%. Due to the high amount of information, we provide detailed numbers on all questions in Tables 5, 6, 7, and 8 in the replication package [46].

5.1 Participants' Demographics

Overall, a total of 947 participants started our survey. Of those, we considered 706 complete and valid responses on Amazon MTurk (101), Prolific (122), Upwork (72), Freelancer (100), Google Play (103), and in the student sample (208) for our analysis. We report the details on the recruitment and participants per platform in Table 1.

Across all platforms, we find that participants reported experience in areas besides development, however, the extent of this varied. Most commonly, developers also had experience as software testers (58.2-70.0%). We found networking skills the most common on Google Play (45.6%) and the least common for Upwork (22.2%) and Prolific participants (21.3%). Reverse engineering knowledge was overall less common, with at most 22.0% on Freelancer. It was least common on Prolific (10.7%) or for students (13.5%). We report the most relevant general demographics in Table 2. Across all samples, our participants predominantly identified themselves as male (83.4%), with CS students (μ: 24.6) followed by Upwork (μ: 27.4) and Prolific (μ: 28.0) being the youngest groups and participants from Google Play (μ: 37.5) the oldest. Most participants (74.8%) studied computer science. However, except for MTurk, where 21.8% self-reported a computer security focus in their studies, at most 4.0% reported this on the other platforms. Furthermore, only a minority of participant reported a disability and only on Freelancer (45.0%) and MTurk (41.6%) we found a large portion to be a primary caregiver for children, elderly or disabled.

Cultural background and language proficiency: Participants across all samples reported a wide variety of ethnicities (cf. Figure 9 in replication package [46]). Overall, we find that on Prolific (86.1%), Google Play (76.7%), MTurk (71.3%), and in the CS student sample (59.6%) most participants reported to be white or of European descent.

English (US) was reported as the native language for a majority of participants on MTurk (78.2%) which is likely due to most of them being located in the U. S. (71.3%), whereas a plurality of participants on Google Play (40.4%) and in the student sample (40.4%) reported German as their native language. Participants on Prolific, Upwork and Freelancer reported a variety of native languages with none of them standing out. Despite only MTurk users being the only sample with a majority of native English speakers, a majority (89.0%) across all platforms reported being very comfortable in answering the survey in English on a five-point Likert scale.

Education and employment: A majority of participants on MTurk (76.2%), Upwork (63.9%) and Freelancer (53.0%) reported having a Bachelor's degree, while the most common degree on Google Play (27.2%) was a master's degree. While not a majority, a Bachelor's degree was most common for students (32.2%) and on Prolific (38.5%). Additionally, a majority within the student sample reported to be students (60.1%) or part-time employees (22.1%) as opposed to other samples,

	MTurk	Prolific	Upwork	Freelancer	Google Play	CS Students
Experience as:						
Reverse Engineer	18 (17.8%)	13 (10.7%)	13 (18.1%)	22 (22.0%)	16 (15.5%)	28 (13.5%)
Network Practitioner	39 (38.6%)	26 (21.3%)	16 (22.2%)	43 (43.0%)	47 (45.6%)	57 (27.4%)
Vulnerability Researcher	38 (37.6%)	22 (18.0%)	16 (22.2%)	28 (28.0%)	36 (35.0%)	34 (16.3%)
Software Tester	60 (59.4%)	81 (66.4%)	49 (68.1%)	70 (70.0%)	60 (58.3%)	121 (58.2%)
Programming/week:						
$<$ 5 hours	8 (7.9%)	17 (13.9%)	6 (8.3%)	4 (4.0%)	4 (3.9%)	25 (12.0%)
5 - 10 hours	23 (22.8%)	25 (20.5%)	10 (13.9%)	11 (11.0%)	16 (15.5%)	55 (26.4%)
10 - 20 hours	23 (22.8%)	28 (23.0%)	9 (12.5%)	6 (6.0%)	19 (18.4%)	47 (22.6%)
20 - 30 hours	21 (20.8%)	20 (16.4%)	18 (25.0%)	16 (16.0%)	24 (23.3%)	34 (16.3%)
30 - 40 hours	14 (13.9%)	18 (14.8%)	14 (19.4%)	27 (27.0%)	21 (20.4%)	22 (10.6%)
$>$ 40 hours	11 (10.9%)	10 (8.2%)	14 (19.4%)	36 (36.0%)	18 (17.5%)	15 (7.2%)
Gender:						
Woman	16 (15.8%)	32 (26.2%)	7 (9.7%)	7 (7.0%)	1 (1.0%)	42 (20.1%)
Man	84 (83.2%)	89 (73.0%)	65 (90.3%)	90 (90.0%)	97 (94.2%)	165 (78.9%)
Non-binary	1 (1.0%)	1 (0.8%)	0 (0.0%)	0 (0.0%)	3 (2.9%)	1 (0.5%)
Age [∅ in years]:	31.6	28.0	27.4	30.1	37.5	24.6
Primary field of study:						
Computer Science	62 (61.4%)	91 (74.6%)	51 (70.8%)	80 (80.0%)	62 (60.2%)	182 (87.5%)
IT security/Cybersecurity	22 (21.8%)	2 (1.6%)	1 (1.4%)	4 (4.0%)	2 (1.9%)	5 (2.4%)
Other eng. disciplines	11 (10.9%)	13 (10.7%)	9 (12.5%)	8 (8.0%)	16 (15.5%)	3 (1.4%)
No of co-workers:						
Solo	15 (14.9%)	48 (39.3%)	31 (43.1%)	47 (47.0%)	41 (39.8%)	67 (32.2%)
In a team	86 (85.1%)	74 (60.7%)	41 (56.9%)	53 (53.0%)	62 (60.2%)	141 (67.8%)

Table 2: Basic demographic information for valid participants across all samples.

where most were full-time employees or self-employed. An exception here was Prolific, where 26.2% of the participants reported to be students, the second-largest group after full-time employees (45.9%). Overall, most participants reported to work as a software developer (78.9%). No other job roles particularly stood out, but overall 34.8% of participants reported to work as an Engineer as a second most common job role. Job role distribution did not vary much between platforms with only students reporting to have roles like data science or ML specialist (20.7%) and academic researcher or scientist (20.2%) more than other platforms.

Workplace size and working hours: Across all platforms, the majority of participants (70.1%) reported working in a small company with less than 500 employees. Additionally, students (μ: 22.7) reported the lowest overall working hours per week. Participants on Freelancer (μ: 39.0) followed by Google Play (μ: 37.0) and MTurk (μ: 36.0) reporting the highest working hours. This stresses how most participant, except for students, who more commonly work part-time, are full-time developers.

Key Points | Participants' Demographics: Software development experience is by far **most common**. Participants are also **(less) experienced** with **security-relevant areas** such as reverse engineering or vulnerability research. **Current job** roles encompass mostly **development**. The **majority** of our participants studied **computer science** with **no security focus,** and many participants work **full-time** in **smaller companies**. **Disabilities** are very **rare**. **Caregiving** (for children) was only common on **MTurk** and **Freelancer**. We find **a wide variety of ethnicity and native languages** between platforms. **Bachelor/Master** degrees are **common**.

5.2 General Programming Experience and Knowledge

In this section, we report our participants' general programming experience and knowledge across all samples.

Development experience: We asked participants to report their total and professional development experience in years (cf. Figure 3). Google Play participants reported both the highest overall (μ: 17.8) and professional development experience (μ: 11.0). In contrast, students were least experienced regarding both overall (μ: 7.0) and professional experience (μ: 2.1). For total programming experience, we find no significant difference, while the professional experience on Google Play is significantly higher than on Upwork (KW, H = 7.43, $p < 0.01$) or for students (KW, H = 11.31, p = 0.03) after correction.

Development tasks per week: Furthermore, we asked participants how many hours they spent on software development tasks per week. The majority of Freelancer (79.0%), Google Play (61.2%) and Upwork (63.8%) participants reported working more than 20 hours per week, confirming the previous findings of their commonly reported full-time status. Conversely, the majority on Prolific (60.7%) and in the student sample (65.8%) and a bit more than half of MTurk participants (54.5%) report that they work 20 hours or less.

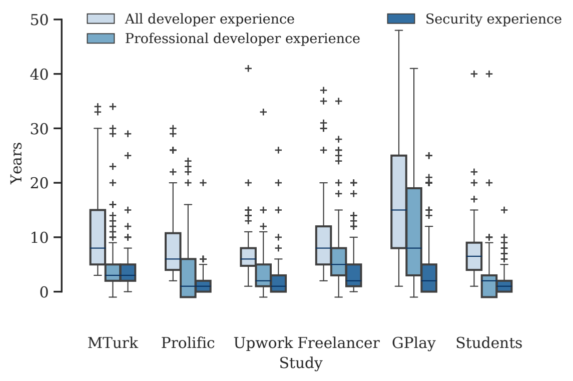
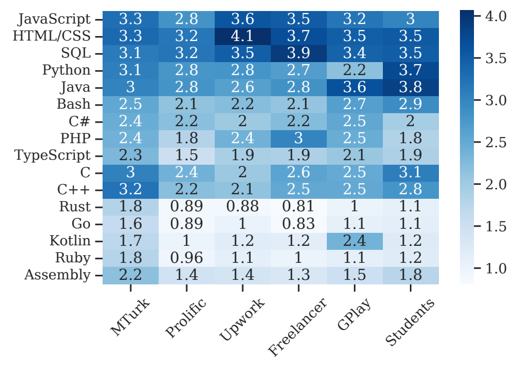
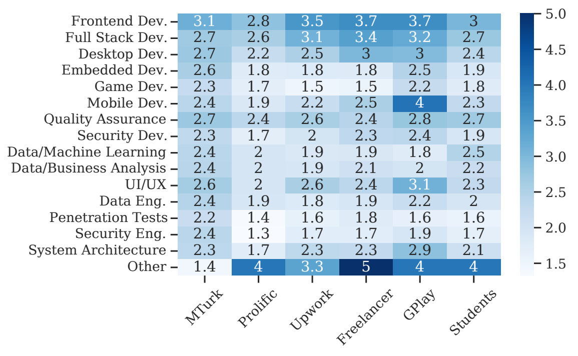


Figure 3: Years of total, professional, and computer security experience for developers across all samples.



(a)



(b)

Figure 4: Developers' average self ratings for (a) proficiency with the top 15 programming, scripting and markup languages (b) proficiency in different development areas

Proficiency in programming, scripting and markup languages: An essential recruitment criteria for developer studies, especially those with practical tasks, is the proficiency with specific programming languages [57, 58, 69] or development areas [82]. Hence, we asked them to rate their proficiencies with the top 15 programming, scripting, or markup languages (cf. Figure 4a), and different development areas (cf. Figure 4b), both taken from the StackOverflow Developer Survey results [10]. Overall, MTurk participants report higher ratings for most programming languages as well as in development areas, although no single language stood out. Proficiency in Rust was rated the lowest in almost all samples, followed by Go and Ruby. While Java was highly rated by both CS students and Google Play developers, likely due to Java being both a popular teaching language, and the basis for Android development, we found SQL to be the best known on Freelancer, and HTML/CSS very highly rated on Upwork.

When examining different development areas, we find that frontend and full stack development were rated the highest overall. This is likely due to them being very broad and widespread areas, which is supported by the StackOverflow Developer Survey, in which they were among the three most commonly stated roles. The average self-rating for mobile development was highest for Google Play participants, notably the highest rating found within the given areas.

Key Points | Proficiency in programming, scripting and markup languages: **Google Play** developers reported the **most** and **student** developers the **least experience in years**. On **Freelancer**, **Google Play**, and **Upwork** we recruited mostly **full-time developers**. **MTurkers** reported the **highest proficiencies** across most **development areas and programming languages**.

5.3 Security Experiences, Knowledge, and Skills

In this section, we provide details on security experience, knowledge and skills our participants reported.

General security experience, knowledge and skills: We asked participants to report their general computer security experience in years, including studying or working (cf. Figure 3). Google Play participants self-reported the highest security experience (μ: 4.4) and Prolific users the lowest (μ: 1.3). However, we found no statistically significant differences between samples.

Based on the SSD-SES scale [79] we found that participants across samples reported different confidence in their secure development skills (Freelancer.com: μ: 26.4; MTurk: μ: 26.0; Upwork: μ: 24.1; GPlay: μ: 22.6; CS Students μ: 21.9; Prolific: μ: 18.8), and the students' confidence to be significantly lower than the other samples (Mann-Whitney U, U = 370–1107, $p < 0.01$).

Implementing specific security features: We were interested in learning our participants' experiences with implementing specific security features, e. g., encryption or storing user credentials (cf. Figure 5). Our participants reported most experience with implementing input validation, authorization and authentication features, using API keys, using encryption

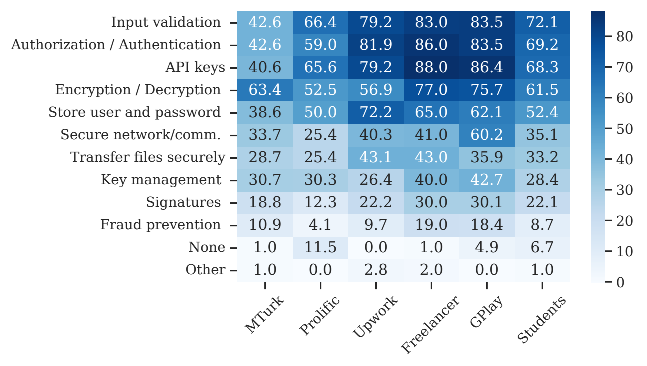


Figure 5: Developers' security features usage.

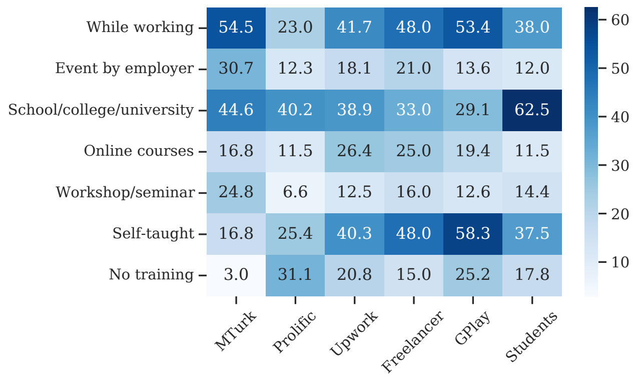


Figure 6: Security related training our participants received.

and storing user credentials. They reported the least experience with cryptographic key management, digital signatures and fraud prevention features. We also found that participants from Upwork, Freelancer and Google Play reported most experience with implementing specific security features. In contrast, participants from MTurk and Prolific reported the lowest experience. CS students were in the middle.

Finding security problems in code: Furthermore, we asked participants how often they used specific techniques to find security problems in their code, including automated or manual code reviews, tools to scan their code for vulnerable libraries and penetration testing.

Overall, we found the usage frequency for these techniques does not differ much between samples (cf. Figure 8 in replication package [46]). For most techniques, participants state to have used them once or occasionally (20.2-47.5%) or not at all (16.9-65.9%). Within all techniques, penetration tests are the least frequent (37.6-65.9% stated to not use or consider them), most likely due to their massive overhead. We find MTurk participants report generally higher frequencies, with 25.7-45.6% stating they use every technique for at least all builds or releases. We find that in general, code reviews are more frequent than other techniques, both manual reviews by other developers (32.1-47.0% at every build or release) or automated code scanning tools (22.6-45.6%).

Receiving security training: We asked participants which security training they received in the past, including training on the job, at school or university, online and self-taught training (cf. Figure 6).

In almost all samples except Prolific, more than one third of participants received a security related training at work, with Google Play (53.4%) and MTurk (54.5%) participants reporting extraordinary high numbers. In contrast, only 10 to 20% of all participants except for MTurk (30.7%) participated in a dedicated security training event organized by their employer. Across all samples, at least roughly a third (29.1% on Google Play) of participants received a security training at educational institutions, the highest being students (62.5%). Online courses were similarly popular across all samples, except for CS students and Prolific, who took them only about half as often as others. While 24.8% of MTurk participants reported to have participated in dedicated workshops or seminars, only 6.6% of Prolific participants and roughly 15% of the remaining participants did. A majority of Google Play participants reported that they were self-taught (58.3%), with similar high numbers on Upwork (40.3%) and Freelancer (48.0%), which fits to the culture of freelancing or self-employment. This is in stark contrast to MTurk participants, where only 16.8% report to be self-taught.

Security-related activities and events: We asked participants about security certifications, participation in security events or CTF contests, software vulnerability disclosure, and submissions to bug bounty programs (cf. Figure 7), as we found these as recruitment criteria or methods in our literature review.

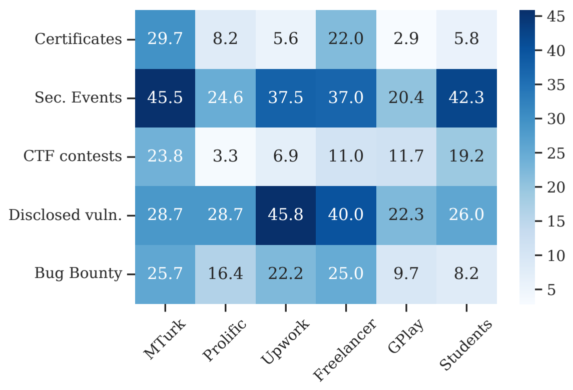


Figure 7: Security related activities and events our participants took part in. The values represent the percentage per sample that stated to have taken part in the respective activity/event.

Overall, we found that participants most commonly attended security related events or had previously disclosed vulnerabilities. However, this did not apply to Google Play developers, where only up to 22.3% participated in any activity, while, e. g., 42.3% of the lesser experienced students stated to have attended security related events. However, we found that *security-related events* were commonly confused with security-related classes by student participants. Again, MTurk participants report surprisingly high numbers in comparison to the other samples, except for disclosing vulnerabilities.

Security in teams vs solo: To examine if security experience differed for participants who worked alone or in a team, we first asked for the number of co-workers in participants' teams. We differentiated between participants who reported no or at least one, regarding the latter as working in a team. Furthermore, the question phrasing in this block slightly differed for both groups, referring either participants' teams or only themselves. Ultimately. we found that the majority of participants (64.7%) on all platforms worked in teams.

Security focus in current field of activity: The majority of participants on Freelancer (52.8%) report a security focus within their teams, while students report this least frequently (26.2%). However, the numbers for having a security focus are overall high, possibly due to a too broad or vague understanding of the term security focus in our question. For solo workers, we find the largest portion (46.7%) of participants with security focus on MTurk, while a lack of security focus is common on Google Play (12.2%), Prolific (10.4%) and for students (9.0%).

Security-relevant tasks: Similarly, we asked participants if they or their team work on security-relevant tasks. A majority of MTurk (54.7%) and a plurality of Freelancer (49.1%) participants report working on security-relevant tasks in a team, while the remaining participants mostly report only partial or no (e. g. 36.9% of students) security-relevant work. For solo workers, a majority on Prolific (54.2%) as well as most on Google Play (34.1%) and in the student sample (49.3%) are not working on security-relevant tasks.

Security-critical decisions: Furthermore, we wanted to know if anyone else (and who) is consulted for security-critical decisions, as these could be very sensitive from an organization's point of view. We find that a majority of participants working in teams across all platforms make security-critical decisions with their team (61.4%). Participants working solo on all platforms mostly make security-critical decisions alone (53.6%).

Security champion[3]*:* Lastly, for the participants working in a team, we wanted to know if they have a security champion in their team. While the majority of participants with teams report to make security-critical decisions with their team, at most 45.3% on MTurk also report to have a security champion, implying a deficit of designated security experts within teams. Overall, we find that attention to security is more frequent for participants working in teams in comparison to those working alone. This might be due to security being regarded as a lesser goal than pure functionality, and therefore seemingly less relevant, especially when the workload cannot be split with other team members.

[3]We define security champion as a co-worker who may not be a trained security expert, but has a large interest and knowledge and can be approached with questions.

Key Points | Security Demographics: **MTurk** along with **Freelancer and Upwork** participants reported the **highest values** for most security related questions. **CS students** and partially **Prolific** participants reported the **lowest**. **Secure development features** were used, but **not by a majority**. **Most common features** included input validation, authorization and authentication, API keys, encryption and storing passwords. **More than a third** of all participants reported **security training** at work. **Security activities** were **not common** in most samples. We found a **security focus and tasks more common in teams** than for solo workers.

6 Limitations

Both the literature survey and the online study have limitations. First, the literature survey is limited to the proceedings listed in Section 3.1. We cross-checked with literature reviews from 2018 [73] and 2021 [47], which also included workshops and lower-ranked venues, finding no significant oversights and believe to have identified and included all relevant papers.

Due to the diverse nature of screening questions, payment types and filtering criteria, we were unable to perfectly replicate all previous work on all tested platforms. We furthermore decided against certain recruitment strategies that were too easily influenced by research team location (e. g., security events or meetups) or personal/professional contacts (e. g., social media, LinkedIn), or that would not allow us to recruit sufficient sample sizes. An exception to this was the student sample. While these are also influenced by location, we argue that they are easy to reach for university researchers, and in our sample, decided to merge several student samples to mitigate the biases.

Participants on Upwork and Freelancer were able to directly communicate with us, and were often interested in additional jobs, which might have been an incentive to report higher scores to leave a better impression, although we do not find both platforms to perform visibly better than others.

For some platforms, we required multiple runs or stops in between due to bureaucratic reasons (i. e., resolving payments with our institutions), resulting in longer data collection time frames for these samples. Furthermore, there may have been instances of the same person participating multiple times through different platforms. However, we asked participants on which recruitment platforms they are active, to which participants reported up to three additional platforms on average (cf. Figure 10 in replication package [46]). We argue that due to other differences in, e. g., ethnic backgrounds, it is unlikely that participants responded multiple times on a large scale.

For the German student sample, due to translations and differing cultural and societal circumstances, participants were able to select a different number of answers for the gender, primary study field and caregiving question. We found no significant impact in the data due to this. Lastly, as recruitment channels change, we believe that this kind of study should be repeated from time to time.

7 Discussion

In this section, we discuss our experiences with the recruitment platforms used in this research and provide recommendations for platforms suitable for certain security study types.

As researchers need to consider their target population when deciding for or against recruitment platforms, we discuss insights from results below that can be helpful for recruitment decisions.

7.1 Participant Characteristics

Development experience, skills and knowledge: Development experience and skills are common selection criteria for secure development studies. Our results offer insights to researchers considering a variety of points along the experience spectrum, depending on their study's needs. We found students to have the least experience while Google Play developers have the highest overall and professional development experience. This finding is also backed up by the fact that most students reported working part-time or studying as their main occupation. In contrast, participants on Google Play and other platforms are older and report more years of experience as well as working full-time, which positively influence their experience. MTurk and Freelancer participants report the second-highest overall and professional development experience, respectively and can be considered reasonable developer recruitment alternatives to Google Play, with the benefit of offering faster recruitment.

Another relevant set of criteria for developer recruitment is the specific skill set participants may have, such as knowledge of a certain programming language or development area which could be useful for studies that want to research, e. g., a security aspect of Python programming. We found MTurk to be the most diverse as participants reported high proficiency levels for most development areas and languages. Google Play developers reported to be more proficient with Java and Kotlin, CS students with Java and Python, while participants on Freelancer and Upwork reported high proficiency with SQL and HTML/CSS. While some of these might be coincidental, it seems sensible to find Android developers to be skilled in Java, and that Java and Python are popular teaching languages for students.

Researchers also often need to balance their experience and skill needs with the amount of recruitment effort required. While we were able to recruit developers on all platforms we sampled, we found that on some, the recruitment was faster or easier. This was especially the case on Prolific and MTurk, as researchers can screen and hire developers in batches, while Upwork and Freelancer are focused on contract-work and therefore require researchers to hire one participant at a time.

Security experience, skills and knowledge: Overall, we found our participants to have experience in various areas of secure development. Regarding experience with security features, we found usage of input validation, authentication, API keys, encryption and password storage most common, especially widespread on Upwork, Freelancer and Google Play, suggesting a wider range of experience with secure development than within other samples. Regarding the frequency of secure development tools to find code problems, MTurk participants reported deploying them on a much more frequent basis than all other samples; we found no meaningful differences between the other samples. While we only regarded participants with development background in our analysis, we asked for experience with several other areas and find security-relevant job roles present in our samples, making recruitment of e. g. reverse engineers on Freelancer a more viable option than on Prolific or Google Play.

Overall, MTurk participants reported the highest values for most security-related questions, suggesting it is easier to recruit developers there. This includes job role diversity, especially in security-related disciplines such as reverse engineering or vulnerability research, as well as years of security experience and their security knowledge confidence, which was rated with the SSD-SES. While MTurk looks promising, we encountered a high number of invalid and very similar answers, likely due to multi-accounting, in our initial run. This required us to redo the sample with an adjusted screening that included programming skill questions from Danilova et al. [30]. We therefore recommend to use MTurk with caution and similar strict screening requirements to increase data quality. While we did not observe this problem on Prolific, we also could only rely on self-reported skill, and decided to redo this platform with the same screener. Future work should investigate our findings in more detail to better assess the recruitment of MTurk users for security development studies. As an alternative, Freelancer and Upwork participants also often reported high values (more than MTurk at some instances) on security-related questions. They are further viable options when researchers are trying to reach a diverse sample. On the other hand, we find students as well as Prolific participants to be less experienced with security, as they reported the least years of security experience, use security features least often, and had the lowest SSD-SES scores. We further found that participants who work in teams are more likely to have a security focus in their job or work on security-relevant tasks when compared to those who work alone. This is another pointer towards using platforms like MTurk, where comparatively more participants work in teams, to recruit security experts.

Sample Diversity. While participant experiences, skills and knowledge are important aspects of recruitment, diversity of demographics might also be essential for research projects. In our survey, Google Play developers report the highest average age and the widest age range, correlating with their high number of working years. CS students are the youngest. Regarding gender, we find an overwhelming majority of male participants over all platforms, which is sadly usual in security research. Only on Prolific (26.2%) and in the student sample (20.1%) we found a comparatively high number of female participants. Regarding ethnicity, we found Freelancer and MTurk to be the most diverse, while Google Play and Prolific resulted in very euro-centric samples. We argue that due to these findings, MTurk and Freelancer offer the widest degree of diversity within their samples.

7.2 Specific Recruitment Strategies

Crowdsourcing: The crowdsourcing platforms MTurk and Prolific offer a straightforward recruitment process. Although both did not offer screening opportunities besides pre-existing and potentially lacking filters, setting up additional screening surveys to find suitable participants is simple. However, we found cases of multi-accounting on MTurk that required us to use specific skill-based screening questions. On both the platforms, it is easy to quickly collect a large number of participants, but control over the sample is limited and dependent on the available filters, which can be insufficient when targetting a specific population. This can also lead to skewed populations, e. g. in case of the very euro-centric Prolific, or influenced by online trends [9]. We recommend MTurk for larger studies with more participants where some noise (i. e. fraud) in the data could be acceptable. Studies with a smaller number of required participants should use platforms like Freelancer or Upwork, where the data is less likely to be noisy. Besides, the slower recruitment should be less of an issue for smaller studies, and researchers can gather more information about the participants' experience via their profiles.

Freelancers: It was more complex to create job postings on Freelancer and Upwork, as these platforms are tailored towards recruiting only a few freelancers for more complex practical tasks such as programming a website, leading to comparatively high minimum wages per job of $10 or $5. Therefore, we needed to provide more information in the setup process about both the job, and our expectations for participants, but were able to directly communicate and sufficiently filter for qualified participants. However, this lead to some users trying to bargain with us which was much more time-consuming than the crowdsourcing platforms. On Upwork, it was possible to include screening questions into the job post. Between the two, we found recruiting on Upwork to be less complex and more intuitive, as well as having benefits such as easier and faster payment options. Additionally, we received only two applications on Freelancer, which forced us to manually contact participants. Therefore, we specifically recommend Upwork to recruit experienced developers for practical coding studies or similarly larger tasks.

Email Invites: We contacted both Google Play developers and students via email. While CS students can be reached using university mailing lists, access to these cannot be taken for granted for all research institutions or unaffiliated researchers. Although we only used public contact information from Google Play, the common practice of harvesting email addresses from GitHub commits is a legal gray area that researchers should be aware of [4]. Additionally, we found that many developer addresses are inactive or lead to ticket systems that might never reach a developer. Overall, distributing a survey via email is the fastest and cheapest, but also offers the worst response rates. As there is no platform for handling payments, offering compensation is more complex. Finally, sending unsolicited email invites is increasingly perceived as spam and the collection and use of non-public contact information is inhibited by emerging privacy laws such as the GDPR in the EU.

> Key Points | Recruitment Advice Developers were present on **all** platforms. **Google Play** developers are the most experienced, but focused on **Java and Android**. Students are **less experienced** than other platforms. While there are developers on **Prolific**, they are **less experienced in security topics**. **Crowdsourcing** platforms should only be used with filtering via **screening questions**, especially **MTurk**. **Freelancing** platforms require a lot of **manual work** and are **more expensive**. While **emails** are cheaper, they have **low response rates**.

8 Conclusion

In this work, we first identified common recruitment strategies for user studies with participants with software development experience. We extracted relevant survey questions from these papers, and designed and tested a questionnaire to study the general and security programming, knowledge, skills, and experience of participants. Finally, we surveyed 706 participants across six samples, and provide detailed insights into their survey responses. Overall, we found that participants across samples varied significantly, and that the characteristics of different recruitment strategies highly influenced their suitability for different study types. In future work, more recruitment platforms we excluded in our work should be investigated for a broader picture of viable recruitment methods for security development studies. Similar, future work could investigate developer samples with a different focus such as system administrators or security engineers. Finally, while we found that MTurk participants self-reported high overall and security experience, skills and knowledge at most instances, this particular finding should be further investigated.

References

[1] Accompanying Project Website. https://publications.teamusec.de/2022-usenix-dev-sampling/.

[2] ACM Conference on Compuer and Communications Security. http://www.sigsac.org/ccs.html (visited on 03/26/2021).

[3] Computer science conference(journal) rank: - aminer. https://www.aminer.org/ranks/conf?category_type=ccf&category=%E7%BD%91%E7%BB%9C%E4%B8%8E%E4%BF%A1%E6%81%AF%E5%AE%89%E5%85%A8&category_en=Network%20and%20Information%20Security (visited on 02/01/2022).

[4] GitHub Privacy Statement. https://docs.github.com/en/github/site-policy/github-privacy-statement (visited on 06/09/2021).

[5] HCI Guidelines for Gender Equity and Inclusivity. https://www.morgan-klaus.com/gender-guidelines.html (visited on 05/09/2021).

[6] Human Factors in Computing Systems. https://chi.acm.org/ (visited on 03/26/2021).

[7] IEEE Symposium on Security and Privacy. https://www.ieee-security.org/TC/SP-Index.html (visited on 03/26/2021).

[8] Network and Distributed Systems Security Symposium. https://www.ndss-symposium.org/ (visited on 03/26/2021).

[9] Prolific | Blog. https://blog.prolific.co/we-recently-went-viral-on-tiktok-heres-what-we-learned/ (visited on 06/09/2021).

[10] StackOverflow 2020 Developer Survey. https://insights.stackoverflow.com/survey/2020 (visited on 03/26/2021).

[11] Symposium on Usable Privacy and Security. https://www.usenix.org/conferences/byname/884 (visited on 03/26/2021).

[12] Top computer science conference ranking | research.com. https://research.com/conference-rankings/computer-science (visited on 02/01/2022).

[13] USENIX Security Symposium. https://www.usenix.org/conferences/byname/108 (visited on 03/26/2021).

[14] Yasemin Acar, Michael Backes, Sascha Fahl, Simson Garfinkel, Doowon Kim, Michelle L. Mazurek, and Christian Stransky. Comparing the Usability of Cryptographic APIs. In *Proc. 38th IEEE Symposium on Security and Privacy (SP'17)*. IEEE, 2017.

[15] Yasemin Acar, Michael Backes, Sascha Fahl, Doowon Kim, Michelle L. Mazurek, and Christian Stransky. You get where you're looking for: The impact of information sources on code security. In *IEEE Symposium on Security and Privacy, SP 2016, San Jose, CA, USA, May 22-26, 2016*, pages 289–305, May 2016.

[16] Yasemin Acar, Christian Stransky, Dominik Wermke, Michelle L. Mazurek, and Sascha Fahl. Security Developer Studies with GitHub Users: Exploring a Convenience Sample. In *Proc. 13th Symposium on Usable Privacy and Security (SOUPS'17)*. USENIX Association, 2017.

[17] Devon Adams, Alseny Bah, Catherine Barwulor, Nureli Musaby, Kadeem Pitkin, and Elissa M Redmiles. Ethics emerging: the story of privacy and security perceptions in virtual reality. In *Fourteenth Symposium on Usable Privacy and Security ({SOUPS} 2018)*, pages 427–442, 2018.

[18] M Ugur Aksu, Enes Altuncu, and Kemal Bicakci. A first look at the usability of openvas vulnerability scanner. In *Workshop on Usable Security (USEC) 2019*. NDSS, 2019.

[19] Noura Alomar, Primal Wijesekera, Edward Qiu, and Serge Egelman. " you've got your nice list of bugs, now what?" vulnerability discovery and management processes in the wild. In *Sixteenth Symposium on Usable Privacy and Security ({SOUPS} 2020)*, pages 319–339, 2020.

[20] Simon Anell, Lea Gröber, and Katharina Krombholz. End user and expert perceptions of threats and potential countermeasures. In *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pages 230–239. IEEE, 2020.

[21] Hala Assal and Sonia Chiasson. Security in the software development lifecycle. In *Fourteenth Symposium on Usable Privacy and Security ({SOUPS} 2018)*, pages 281–296, 2018.

[22] Louise M Axon, Bushra Alahmadi, Jason RC Nurse, Michael Goldsmith, and Sadie Creese. Sonification in security operations centres: what do security practitioners think? *arXiv preprint arXiv:1807.06706*, 2018.

[23] Steffen Becker, Carina Wiesen, Nils Albartus, Nikol Rummel, and Christof Paar. An exploratory study of hardware reverse engineering—technical and cognitive processes. In *Sixteenth Symposium on Usable Privacy and Security ({SOUPS} 2020)*, pages 285–300, 2020.

[24] Frank R Bentley, Nediyana Daskalova, and Brooke White. Comparing the reliability of amazon mechanical turk and survey monkey to traditional market research surveys. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, pages 1092–1099, 2017.

[25] Xander Bouwman, Harm Griffioen, Jelle Egbers, Christian Doerr, Bram Klievink, and Michel van Eeten. A different cup of {TI}? the added value of commercial threat intelligence. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*, pages 433–450, 2020.

[26] Karoline Busse, Julia Schäfer, and Matthew Smith. Replication: no one can hack my mind revisiting a study on expert and non-expert security practices and advice. In *Fifteenth Symposium on Usable Privacy and Security ({SOUPS} 2019)*, 2019.

[27] George Chalhoub, Ivan Flechais, Norbert Nthala, and Ruba Abu-Salma. Innovation inaction or in action? the role of user experience in the security and privacy design of smart home cameras. In *Sixteenth Symposium on Usable Privacy and Security ({SOUPS} 2020)*, pages 185–204, 2020.

[28] Jesse Chandler, Cheskie Rosenzweig, Aaron J Moss, Jonathan Robinson, and Leib Litman. Online panels in social science research: Expanding sampling methods beyond mechanical turk. *Behavior research methods*, 51(5):2022–2038, 2019.

[29] Anastasia Danilova, Alena Naiakshina, Johanna Deuter, and Matthew Smith. Replication: On the ecological validity of online security developer studies: Exploring deception in a password-storage study with freelancers. In *Sixteenth Symposium on Usable Privacy and Security ({SOUPS} 2020)*, pages 165–183, 2020.

[30] Anastasia Danilova, Alena Naiakshina, Stefan Horstmann, and Matthew Smith. Do you really code? designing and evaluating screening questions for online surveys with programmers. *arXiv preprint arXiv:2103.04429*, 2021.

[31] Alexander De Luca, Sauvik Das, Martin Ortlieb, Iulia Ion, and Ben Laurie. Expert and non-expert attitudes towards (secure) instant messaging. In *Twelfth Symposium on Usable Privacy and Security ({SOUPS} 2016)*, pages 147–157, 2016.

[32] Erik Derr, Sven Bugiel, Sascha Fahl, Yasemin Acar, and Michael Backes. Keep me updated: An empirical study of third-party library updatability on android. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 2187–2200, 2017.

[33] Constanze Dietrich, Katharina Krombholz, Kevin Borgolte, and Tobias Fiebig. Investigating system operators' perspective on security misconfigurations. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 1272–1289, 2018.

[34] Pardis Emami-Naeini, Yuvraj Agarwal, Lorrie Faith Cranor, and Hanan Hibshi. Ask the experts: What should be on an iot privacy and security label? In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 447–464. IEEE, 2020.

[35] Ksenia Ermoshina, Harry Halpin, and Francesca Musiani. Can johnny build a protocol? co-ordinating developer and user intentions for privacy-enhanced secure messaging protocols. In *European Workshop on Usable Security*, 2017.

[36] Felix Fischer, Huang Xiao, Ching-Yu Kao, Yannick Stachelscheid, Benjamin Johnson, Danial Razar, Paul Fawkesley, Nat Buckley, Konstantin Böttinger, Paul Muntean, and Jens Grossklags. Stack Overflow Considered Helpful! Deep Learning Security Nudges Towards Stronger Cryptography. In *Proc. 28th Usenix Security Symposium (SEC'19)*. USENIX Association, 2019.

[37] Alexander Gamero-Garrido, Stefan Savage, Kirill Levchenko, and Alex C Snoeren. Quantifying the pressure of legal risks on third-party vulnerability research. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1501–1513, 2017.

[38] Yossi Gilad, Avichai Cohen, Amir Herzberg, Michael Schapira, and Haya Shulman. Are we there yet? on rpki's deployment and security. In *NDSS*, 2017.

[39] Peter Leo Gorski, Yasemin Acar, Luigi Lo Iacono, and Sascha Fahl. Listen to developers! a participatory design study on security warnings for cryptographic apis. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2020.

[40] Peter Leo Gorski, Luigi Lo Iacono, Dominik Wermke, Christian Stransky, Sebastian Möller, Yasemin Acar, and Sascha Fahl. Developers deserve security warnings, too: On the effect of integrated security advice on cryptographic {API} misuse. In *Fourteenth Symposium on Usable Privacy and Security ({SOUPS} 2018)*, pages 265–281, 2018.

[41] Julie M Haney and Wayne G Lutters. " it's scary… it's confusing… it's dull": How cybersecurity advocates overcome negative perceptions of security. In *Fourteenth Symposium on Usable Privacy and Security ({SOUPS} 2018)*, pages 411–425, 2018.

[42] Julie M Haney, Mary Theofanos, Yasemin Acar, and Sandra Spickard Prettyman. "we make it a big deal in the company": Security mindsets in organizations that develop cryptographic products. In *Fourteenth Symposium on Usable Privacy and Security ({SOUPS} 2018)*, pages 357–373, 2018.

[43] Norman Hänsch, Andrea Schankin, Mykolai Protsenko, Felix Freiling, and Zinaida Benenson. Programming experience might not help in comprehending obfuscated source code efficiently. In *Fourteenth Symposium on Usable Privacy and Security ({SOUPS} 2018)*, pages 341–356, 2018.

[44] Mokter Hossain. Users' motivation to participate in online crowdsourcing platforms. In *2012 International Conference on Innovation Management and Technology Research*, pages 310–315. IEEE, 2012.

[45] Luigi Lo Iacono and Peter Leo Gorski. I do and i understand. not yet true for security apis. so sad. In *Proc. of the 2nd European Workshop on Usable Security, ser. EuroUSEC*, volume 17, 2017.

[46] Harjot Kaur, Sabrina Amft, Daniel Votipka, Yasemin Acar, and Sascha Fahl. Replication Package: "Where to Recruit for Security Development Studies: Comparing Six Software Developer Samples". `https://doi.org/10.25835/wg7xhqmh`, 2022.

[47] Mannat Kaur, Michel van Eeten, Marijn Janssen, Kevin Borgolte, and Tobias Fiebig. Human factors in security research: Lessons learned from 2008-2018. *arXiv preprint arXiv:2103.13287*, 2021.

[48] Aniket Kittur, Ed H Chi, and Bongwon Suh. Crowdsourcing user studies with mechanical turk. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 453–456, 2008.

[49] Katharina Krombholz, Karoline Busse, Katharina Pfeffer, Matthew Smith, and Emanuel von Zezschwitz. "if https were secure, i wouldn't need 2fa" – end user and administrator mental models of https. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 246–263. IEEE, 2019.

[50] Katharina Krombholz, Wilfried Mayer, Martin Schmiedecker, and Edgar R. Weippl. "i have no idea what i'm doing" - on the usability of deploying HTTPS. In Engin Kirda and Thomas Ristenpart, editors, *26th USENIX Security Symposium, USENIX Security 2017, Vancouver, BC, Canada, August 16-18, 2017*, pages 1339–1356. USENIX Association, 2017.

[51] Frank Li, Lisa Rogers, Arunesh Mathur, Nathan Malkin, and Marshini Chetty. Keepers of the machines: Examining how system administrators manage software updates for multiple machines. In *Fifteenth Symposium on Usable Privacy and Security ({SOUPS} 2019)*, 2019.

[52] Abraham H Mhaidli, Yixin Zou, and Florian Schaub. " we can't live without them!" app developers' adoption of ad networks and their considerations of consumer risks. In *Fifteenth Symposium on Usable Privacy and Security ({SOUPS} 2019)*, 2019.

[53] Ola Aleksandra Michalec, Dirk van der Linden, Sveta Milyaeva, and Awais Rashid. Industry responses to the european directive on security of network and information systems (nis): Understanding policy implementation practices across critical infrastructures. In *Sixteenth Symposium on Usable Privacy and Security ({SOUPS} 2020)*, pages 301–317, 2020.

[54] Dongliang Mu, Alejandro Cuevas, Limin Yang, Hang Hu, Xinyu Xing, Bing Mao, and Gang Wang. Understanding the reproducibility of crowd-reported security vulnerabilities. In *27th {USENIX} Security Symposium ({USENIX} Security 18)*, pages 919–936, 2018.

[55] Alena Naiakshina, Anastasia Danilova, Eva Gerlitz, and Matthew Smith. On conducting security developer studies with cs students: Examining a password-storage study with cs students, freelancers, and company developers. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2020.

[56] Alena Naiakshina, Anastasia Danilova, Eva Gerlitz, Emanuel von Zezschwitz, and Matthew Smith. " if you want, i can store the encrypted password" a password-storage field study with freelance developers. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–12, 2019.

[57] Alena Naiakshina, Anastasia Danilova, Christian Tiefenau, Marco Herzog, Sergej Dechand, and Matthew Smith. Why do developers get password storage wrong? a qualitative usability study. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 311–328, 2017.

[58] Alena Naiakshina, Anastasia Danilova, Christian Tiefenau, and Matthew Smith. Deception task design in developer password studies: Exploring a student sample. In *Fourteenth Symposium on Usable Privacy and Security ({SOUPS} 2018)*, pages 297–313, 2018.

[59] Duc-Cuong Nguyen, Dominik Wermke, Yasemin Acar, Michael Backes, Charles Weir, and Sascha Fahl. A stitch in time: Supporting android developers in writing secure code. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, pages 1065–1077, Oct 2017.

[60] Timothy Nosco, Jared Ziegler, Zechariah Clark, Davy Marrero, Todd Finkler, Andrew Barbarello, and W Michael Petullo. The industrial age of hacking. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*, pages 1129–1146, 2020.

[61] Daniela Seabra Oliveira, Tian Lin, Muhammad Sajidur Rahman, Rad Akefirad, Donovan Ellis, Eliany Perez, Rahul Bobhate, Lois A DeLong, Justin Cappos, and Yuriy Brun. {API} blindspots: Why experienced developers write vulnerable code. In *Fourteenth Symposium on Usable Privacy and Security ({SOUPS} 2018)*, pages 315–328, 2018.

[62] Eyal Peer, Laura Brandimarte, Sonam Samat, and Alessandro Acquisti. Beyond the turk: Alternative platforms for crowdsourcing behavioral research. *Journal of Experimental Social Psychology*, 70:153–163, 2017.

[63] Stanley Presser, Mick P. Couper, Judith T. Lessler, Elizabeth Martin, Jean Martin, Jennifer M. Rothgeb, and Eleanor Singer. Methods for testing and evaluating survey questions. *Public Opinion Quarterly*, 68(1):109–130, March 2004.

[64] Elissa M Redmiles, Sean Kross, and Michelle L Mazurek. How well do my results generalize? comparing security and privacy survey results from mturk, web, and telephone samples. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 1326–1343. IEEE, 2019.

[65] Elissa M Redmiles, Noel Warford, Amritha Jayanti, Aravind Koneru, Sean Kross, Miraida Morales, Rock Stevens, and Michelle L Mazurek. A comprehensive quality evaluation of security and privacy advice on the web. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*, pages 89–108, 2020.

[66] Andrew Ruef, Michael Hicks, James Parker, Dave Levin, Michelle L Mazurek, and Piotr Mardziel. Build it, break it, fix it: Contesting secure development. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 690–703, 2016.

[67] Sebastian Schnorf, Aaron Sedley, Martin Ortlieb, and Allison Woodruff. A comparison of six sample providers regarding online privacy benchmarks. 2014.

[68] Adam Shostack, Matthew Smith, Sam Weber, and Mary Ellen Zurko. Empirical Evaluation of Secure Development Processes (Dagstuhl Seminar 19231). *Dagstuhl Reports*, 9(6):1–25, 2019.

[69] Justin Smith, Lisa Nguyen Quang Do, and Emerson Murphy-Hill. Why can't johnny fix vulnerabilities: A usability evaluation of static analysis tools for security. In *Sixteenth Symposium on Usable Privacy and Security ({SOUPS} 2020)*, pages 221–238, 2020.

[70] Rock Stevens, Daniel Votipka, Elissa M Redmiles, Colin Ahern, Patrick Sweeney, and Michelle L Mazurek. The battle for new york: a case study of applied digital threat modeling at the enterprise level. In *27th {USENIX} Security Symposium ({USENIX} Security 18)*, pages 621–637, 2018.

[71] Ben Stock, Giancarlo Pellegrino, Frank Li, Michael Backes, and Christian Rossow. Didn't you hear me?—towards more successful web vulnerability notifications. 2018.

[72] Kathryn T Stolee and Sebastian Elbaum. Exploring the use of crowdsourcing to support empirical studies in software engineering. In *Proceedings of the 2010 ACM-IEEE international symposium on Empirical software engineering and measurement*, pages 1–4, 2010.

[73] Mohammad Tahaei and Kami Vaniea. A Survey on Developer-Centred Security. In *Proc. 2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE, 2019.

[74] Mary Theofanos, Brian Stanton, Susanne Furman, Sandra Spickard Prettyman, and Simson Garfinkel. Be prepared: How us government experts think about cybersecurity. In *Workshop on Usable Security (USec). Internet Society*, 2017.

[75] Tyler W Thomas, Madiha Tabassum, Bill Chu, and Heather Lipford. Security during application development: An application security expert perspective. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–12, 2018.

[76] Christian Tiefenau, Maximilian Häring, Katharina Krombholz, and Emanuel von Zezschwitz. Security, availability, and multiple information sources: Exploring update behavior of system administrators. In *Sixteenth Symposium on Usable Privacy and Security ({SOUPS} 2020)*, pages 239–258, 2020.

[77] Christian Tiefenau, Emanuel von Zezschwitz, Maximilian Häring, Katharina Krombholz, and Matthew Smith. A usability evaluation of let's encrypt and certbot: Usable security done right. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 1971–1988, 2019.

[78] Artem Voronkov, Leonardo A Martucci, and Stefan Lindskog. System administrators prefer command line interfaces, don't they? an exploratory study of firewall interfaces. In *Fifteenth Symposium on Usable Privacy and Security ({SOUPS} 2019)*, 2019.

[79] Daniel Votipka, Desiree Abrokwa, and Michelle L Mazurek. Building and validating a scale for secure software development self-efficacy. In *Proc. CHI Conference on Human Factors in Computing Systems (CHI'20)*. ACM, 2020.

[80] Daniel Votipka, Kelsey R Fulton, James Parker, Matthew Hou, Michelle L Mazurek, and Michael Hicks. Understanding security mistakes developers make: Qualitative analysis from build it, break it, fix it. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*, pages 109–126, 2020.

[81] Daniel Votipka, Seth Rabin, Kristopher Micinski, Jeffrey S Foster, and Michelle L Mazurek. An observational investigation of reverse engineers' processes. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*, pages 1875–1892, 2020.

[82] Daniel Votipka, Rock Stevens, Elissa Redmiles, Jeremy Hu, and Michelle Mazurek. Hackers vs. Testers: A Comparison of Software Vulnerability Discovery Processes. In *Proc. 39th IEEE Symposium on Security and Privacy (SP'18)*. IEEE, 2018.

[83] Charles Weir, Ben Hermann, and Sascha Fahl. From needs to actions to secure apps? the effect of requirements and developer practices on app security. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*, pages 289–305, 2020.

[84] Charles Weir, Awais Rashid, and James Noble. I'd like to have an argument, please: Using dialectic for effective app security. 2017.

[85] Dominik Wermke, Nicolas Huaman, Yasemin Acar, Brad Reaves, Patrick Traynor, and Sascha Fahl. A Large Scale Investigation of Obfuscation Use in Google Play. In *Proc. 34th Annual Computer Security Applications Conference (ACSAC'18)*. ACM, 2018.

[86] Flynn Wolf, Ravi Kuber, and Adam J Aviv. " pretty close to a must-have" balancing usability desire and security concern in biometric adoption. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–12, 2019.

[87] Khaled Yakdan, Sergej Dechand, Elmar Gerhards-Padilla, and Matthew Smith. Helping johnny to analyze malware: A usability-optimized decompiler and malware analysis user study. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 158–177. IEEE, 2016.

A Appendix: Literature survey tables

Meta Data						Recruitment Strategies																								
Venue	**Year**	**Cite**	**Participants**	**#Participants**	**Compensation**	**GitHub**	**Google Play**	**CS Students**	**Contacts**	**Snowball**	**Sec. Events**	**Companies**	**Meetups**	**Flyers**	**Twitter**	**FB Groups**	**FB Ads**	**Slack**	**Reddit**	**Forums**	**Mailing Lists**	**Blogs**	**MTurk**	**Prolific**	**Upwork**	**Freelancer**	**Craigslist**	**LinkedIn**	**Xing**	**Others**
CCS	2016	[66]	O	301	-			●			●																			
	2017	[59]	D	40	P		●	●	●																					
	2017	[32]	D	203	-		●																							
	2017	[37]	E	114	N				●												●							●		●
	2017	[57]	D	20	P			●						●							●									
	2018	[33]	A	227	-						●	●									●									●
	2019	[77]	A	31	P			●													●									
CHI	2018	[75]	E	32	P				●	●	●																			
	2019	[56]	D	43	P																					●				
	2019	[86]	E	19	N				●		●																			
	2020	[79]	D, E	333	P, R				●			●	●					●	●	●				●	●			●		
	2020	[39]	D	25	N				●	●		●									●									
	2020	[55]	D	36	P				●	●		●																	●	
EuroUSEC	2017	[35]	D, E	21	N	●			●		●																			
	2017	[45]	D	55	N				●			●																		
	2017	[84]	E	12	N				●																					
	2020	[20]	E	8	R				●																					●
NDSS	2017	[38]	A, E	>100	N																●									
	2018	[71]	A	193	N	●																								●
S&P	2016	[87]	E	30	P			●				●																		
	2016	[15]	D	349	P		●	●					●								●						●			
	2017	[14]	D	256	-	●																								
	2018	[82]	E	25	N				●	●		●	●																	●
	2019	[49]	A	12	P				●			●									●									●
	2020	[34]	E	22	P				●	●		●																		
SOUPS	2016	[31]	E	16	P																									●
	2017	[16]	D	307	-	●																								
	2018	[41]	E	28	-				●	●		●																		●
	2018	[42]	O	21	N				●		●																			
	2018	[61]	D	109	P			●	●	●		●		●	●		●			●								●		
	2018	[58]	D	40	P			●						●							●									
	2018	[40]	D	53	-	●														●	●									
	2018	[17]	D	10	P												●		●	●										
	2018	[43]	D	66	P			●						●							●									
	2018	[21]	D	13	P				●											●										●
	2019	[52]	D	59	P, R		●		●				●			●			●								●	●		
	2019	[51]	A	119	P, R						●								●		●	●								●
	2019	[26]	E	75	N				●	●	●				●				●		●									
	2019	[78]	A	300	-				●										●	●	●									
	2020	[19]	E	53	R				●			●			●			●												
	2020	[23]	E	9	P			●	●																					
	2020	[53]	E	30	N				●	●	●																			●
	2020	[27]	O	20	N					●		●		●														●		
	2020	[29]	D	43	P																					●				
	2020	[76]	A	74	R				●		●				●				●	●										
	2020	[69]	D	12	N																									
USEC	2017	[74]	E	23	-							●																		
	2018	[22]	E	41	N				●																					
	2019	[18]	E	10	-							●																		
USENIX	2017	[50]	A, E	35	N			●			●	●																		
	2018	[70]	A, O	25	-							●																		
	2018	[54]	E, O	43	N							●																		
	2019	[36]	D	27	P																●									
	2020	[65]	E	41	PP				●						●							●						●		
	2020	[25]	E	14	N				●																			●		
	2020	[81]	D	16	P				●		●	●							●	●										
	2020	[83]	D	335	N		●																							
	2020	[60]	O	12	N									●																●
	2020	[80]	O	94	N						●																			

Table 3: Paper collection after evaluation of previous expert studies in usable security research.
Participants: O = Other, D = Dev, A = Admin, E = Expert; **Compensation:** P = Payment, PP = Performance based, R = Raffle, - = None, N = Not Explicitly Mentioned

Venue	Cite	Author	Papers
CCS	[66]	Ruef et al.	Build It, Break It, Fix It: Contesting Secure Development
	[59]	Nguyen et al.	A Stitch in Time: Supporting Android Developers in WritingSecure Code
	[32]	Derr et al.	Keep me Updated: An Empirical Study of Third-Party Library Updatability on Android
	[37]	Gamero-Garrido et al.	Quantifying the Pressure of Legal Risks on Third-party Vulnerability Research
	[57]	Naiakshina et al.	Why Do Developers Get Password Storage Wrong? A Qualitative Usability Study
	[33]	Dietrich et al.	Investigating System Operators' Perspective on Security Misconfigurations
	[77]	Tiefenau et al.	A Usability Evaluation of Let's Encrypt and Certbot: Usable Security Done Right
CHI	[75]	Thomas et al.	Security During Application Development: an Application Security Expert Perspective
	[56]	Naiakshina et al.	"If you want, I can store the encrypted password": A Password-Storage Field Study with Freelance Developers
	[86]	Wolf et al.	"Pretty Close to a Must-Have": Balancing Usability Desire and Security Concern in Biometric Adoption
	[79]	Votipka et al.	Building and Validating a Scale for Secure Software Development Self-Efficacy
	[39]	Gorski et al.	Listen to Developers! A Participatory Design Study on Security Warnings for Cryptographic APIs
	[55]	Naiakshina et al.	On Conducting Security Developer Studies with CS Students: Examining a Password-Storage Study with CS Students, Freelancers, and Company Developers
EuroUSEC	[35]	Ermoshina et al.	Can Johnny Build a Protocol? Co-ordinating developer and user intentions for privacy-enhanced secure messaging protocols
	[45]	Iacono and Gorski	I Do and I Understand. Not Yet True for Security APIs. So Sad.
	[84]	Weir et al.	I'd Like to Have an Argument, Please: Using Dialectic for Effective App Security
	[20]	Anell et al.	End User and Expert Perceptions of Threats and Potential Countermeasures
NDSS	[38]	Gilad et al.	Are We There Yet? On RPKI's Deployment and Security
	[71]	Stock et al.	Didn't You Hear Me? — Towards More Successful Web Vulnerability Notifications
S&P	[87]	Yakdan et al.	Helping Johnny to Analyze Malware: A Usability-Optimized Decompiler and Malware Analysis User Study
	[15]	Acar et al.	You Get Where You're Looking for: The Impact of Information Sources on Code Security
	[14]	Acar et al.	Comparing the Usability of Cryptographic APIs
	[82]	Votipka et al.	Hackers vs. Testers: A Comparison of Software Vulnerability Discovery Processes
	[49]	Krombholz et al.	"If HTTPS Were Secure, I Wouldn't Need 2FA"- End User and Administrator Mental Models of HTTPS
	[34]	Emami-Naeini et al.	Ask the Experts: What Should Be on an IoT Privacy and Security Label?
SOUPS	[31]	De Luca et al.	Expert and Non-Expert Attitudes towards (Secure) Instant Messaging
	[16]	Acar et al.	Security Developer Studies with GitHub Users: Exploring a Convenience Sample
	[41]	Haney et al.	"It's Scary. . . It's Confusing. . . It's Dull": How Cybersecurity Advocates Overcome Negative Perceptions of Security
	[42]	Haney et al.	"We make it a big deal in the company": Security Mindsets in Organizations that Develop Cryptographic Products
	[61]	Oliveira et al.	API Blindspots: Why Experienced Developers Write Vulnerable Code
	[58]	Naiakshina et al.	Deception Task Design in Developer Password Studies: Exploring a Student Sample
	[40]	Gorski et al.	Developers Deserve Security Warnings, Too: On the Effect of Integrated Security Advice on Cryptographic API Misuse
	[17]	Adams et al.	Ethics Emerging: the Story of Privacy and Security Perceptions in Virtual Reality
	[43]	Hänsch et al.	Programming Experience Might Not Help in Comprehending Obfuscated Source Code Efficiently
	[21]	Assal and Chiasson	Security in the Software Development Lifecycle
	[52]	Mhaidli et al.	"We Can't Live Without Them!" App Developers' Adoption of Ad Networks and Their Considerations of Consumer Risks
	[51]	Li et al.	Keepers of the Machines: Examining How System Administrators Manage Software Updates for Multiple Machines
	[26]	Busse et al.	Replication: No One Can Hack My Mind Revisiting a Study on Expert and Non-Expert Security Practices and Advice
	[78]	Voronkov et al.	System Administrators Prefer Command Line Interfaces, Don't They? An Exploratory Study of Firewall Interfaces
	[19]	Alomar et al.	"You've Got Your Nice List of Bugs, Now What?" Vulnerability Discovery and Management Processes in the Wild
	[23]	Becker et al.	An Exploratory Study of Hardware Reverse Engineering — Technical and Cognitive Processes
	[53]	Michalec et al.	Industry Responses to the European Directive on Security of Network and Information Systems (NIS): Understanding policy implementation practices across critical infrastructures
	[27]	Chalhoub et al.	Innovation Inaction or In Action? The Role of User Experience in the Security and Privacy Design of Smart Home Cameras
	[29]	Danilova et al.	Replication: On the Ecological Validity of Online Security Developer Studies: Exploring Deception in a Password-Storage Study with Freelancers
	[76]	Tiefenau et al.	Security, Availability, and Multiple Information Sources: Exploring Update Behavior of System Administrators
	[69]	Smith et al.	Why Can't Johnny Fix Vulnerabilities: A Usability Evaluation of Static Analysis Tools for Security
USEC	[74]	Theofanos et al.	Be Prepared: How US Government Experts Think About Cybersecurity
	[22]	Axon et al.	Sonification in security operations centres: what do security practitioners think?
	[18]	Aksu et al.	A First Look at the Usability of OpenVAS Vulnerability Scanner
USENIX	[50]	Krombolz et al.	"I Have No Idea What I'm Doing" - On the Usability of Deploying HTTPS
	[70]	Stevens et al.	The Battle for New York: A Case Study of Applied Digital Threat Modeling at the Enterprise Level
	[54]	Mu et al.	Understanding the Reproducibility of Crowd-reported Security Vulnerabilities
	[36]	Fischer et al.	Stack Overflow Considered Helpful! Deep Learning Security Nudges Towards Stronger Cryptography
	[65]	Redmiles et al.	A Comprehensive Quality Evaluation of Security and Privacy Advice on the Web
	[25]	Bouwman et al.	A different cup of TI? The added value of commercial threat intelligence
	[81]	Votipka et al.	An Observational Investigation of Reverse Engineers' Processes
	[83]	Weir et al.	From Needs to Actions to Secure Apps? The Effect of Requirements and Developer Practices on App Security
	[60]	Nosco et al.	The Industrial Age of Hacking
	[80]	Votipka et al.	Understanding security mistakes developers make: Qualitative analysis from Build It, Break It, Fix It

Table 4: List of papers corresponding to the papers mentioned in Table 3.