Crowdsourcing Software Vulnerability Discovery: Models, Dimensions, and Directions

Mortada Al-Banna[1](✉), Boualem Benatallah[1], Moshe C. Barukh[1], Elisa Bertino[2], and Salil Kanhere[1]

[1] UNSW, Sydney, Australia
m.al-banna@unsw.edu.au
[2] Purdue University, West Lafayette, USA

Abstract. Software systems prove indispensable amongst a variety of fields. With our increasing reliance on them coupled with their heightened complexity, the demand for protection increases as well. In this article, we explore how crowdsourcing could be used for vulnerability discovery. We examine the models of crowdsourcing that has been applied in vulnerability discovery, identify dimensions of this crowdsourced task, and discuss applicable concerns and future research directions.

Keywords: Crowdsourcing · Security · Vulnerability discovery · Bug-bounty

1 Introduction

A vulnerability is a security flaw that arises from system design, implementation, or maintenance (e.g., SQL injection vulnerability, and memory corruption vulnerability). By exploiting these vulnerabilities, malicious parties could gain unauthorized access to protected resources. Software systems are becoming increasingly complex with modern-day development distributed across multiple heterogeneous, autonomous, and evolving cloud services. Furthermore, the reliance on third-party software (e.g., cloud, open-APIs, and external libraries) make it difficult for in-house IT experts to deal with inherent risks of using external software. To overcome vulnerability issues, organizations rely on several methods (e.g., automated tools, penetration testing firms, and crowdsourced vulnerability discovery). In this paper, we specifically examine how vulnerability management can be improved with crowdsourcing.

Crowdsourcing harnesses the wisdom of large communities working independently to solve problems, much as open source does for software development. Several crowdsourcing platforms have emerged like Crowdflower, Innocentive, TopCoder, Kaggle, and uTest. Some platforms use crowdsourcing for tasks that require skilled workers (e.g., Web design, testing, Web development tasks, and R&D challenges).

In a crowdsourced vulnerability discovery program (also known as a bug bounty), software providers (who could be the task requester here) submit vulnerability discovery tasks to a community of security professionals (henceforth referred to as SecPros).

W. Zhang et al. (Eds.): WISE 2021, LNCS 13080, pp. 3–13, 2021.
https://doi.org/10.1007/978-3-030-90888-1_1

SecPros play the role of the attackers and compete to discover and report vulnerabilities. Software providers then evaluate the outcomes and may reward SecPros that submit valid vulnerabilities. Rewards can be monetary, gifts, reputation badges, etc. Crowdsourcing vulnerability discovery could be considered economical compared to hiring full-time SecPros or relying on penetration testing firms, giving its higher volume of output in a shorter time [1]. It has been observed that malicious parties are more agile in exploiting vulnerabilities than software providers are in repairing them[1]. It thus makes sense to harness the power of a crowd of SecPros to discover vulnerabilities as quickly as possible following what Linus' Law stated, "given enough eyeballs, all bugs are shallow" [2]. Additionally, diversity in expertise of SecPros is also highly desired. Empirical studies have confirmed that SecPros with different backgrounds, skills and interests tend to discover different vulnerabilities [3].

Previous research has investigated crowdsourcing models and dimensions in general areas like software engineering [4]. Crowdsourcing vulnerability discovery differ from general crowdsourcing in the following (i) tasks are open-ended, as the requester does not know how many vulnerabilities will be discovered (ii) although it is competition based crowdsourcing but there may be multiple winners; (iii) the crowd participants need to possess very specific and specialized skills compared to participants in general crowd tasks; (iv) tasks (and data) are by nature very sensitive and require special agreements regarding privacy, intellectual property and data protection; (v) some commonly used quality assurance techniques, such as redundancy or voting, do not make sense, as information about vulnerabilities are sensitive and remains so until the vulnerability is remediated; and (vi) rewards are different since there might be different tiers for rewards tailored according to the severity of the vulnerability (e.g., high severity vulnerabilities pay higher reward). Previous research has found that the lack of knowledge to manage the task is one of the pre-adoption concerns that organizations have in regard to crowdsourcing vulnerability discovery [5]. This article provides an understanding of crowdsourced vulnerability discovery. We identify crowdsourcing models and key dimensions to characterize vulnerability discovery tasks. We also discuss future research directions, that we believe will help in the advancement of the field.

2 Models for Crowdsourcing Vulnerability Discovery

The concept of crowdsourcing vulnerability discovery is relatively new. NetScape in 1995 launched the first official program[2]. Since then, several big companies (e.g., Google, Facebook, and Microsoft) have adopted the same concept and launched their own programs. Moreover, several targeted crowdsourcing platforms for vulnerability discovery have been established such as BugCrowd (bugcrowd.com), Cobalt (cobalt.io), and Synack (synack.com). Additionally, cyber security competitions are regularly held (Pwn2Own, Pwnium, and Pwn0rama). In the following, we examine the models that involve responsible disclosure of vulnerabilities. We do not consider unregulated markets for vulnerabilities (e.g., Black, or Grey markets).

[1] https://www.cyber.gov.au/acsc/view-all-content/reports-and-statistics/acsc-annual-cyber-threat-report-july-2019-june-2020.

[2] www.theregister.co.uk/2016/02/22/bug_bounty_feature.

2.1 Direct Vulnerability Discovery

In this model, the organization (software vendor or provider) makes an open call, describes the scope and rules of the task. SecPros can then perform vulnerability testing and submit a report describing the vulnerabilities they have discovered. The organization then verifies the legitimacy of the submitted vulnerability reports and determines the proper reward if any. In this model, there is no time frame for accepting submissions (i.e., open duration), unless the organization decides to terminate the program or migrate to another model. In this model, organization is fully responsible for managing the crowdsourced vulnerability discovery program.

An example for direct vulnerability discovery programs is Mozilla bug bounty program[3]. They provide monetary rewards ranging from 500 USD for medium vulnerabilities to more than 10,000 USD for exceptional vulnerabilities. Mozilla also provides non-monetary incentives such as expressing gratitude on the hall of fame. Mozilla SecPros submit vulnerability reports directly to Mozilla and the Bounty Committee assesses the legitimacy of the submission, determines the severity of the vulnerability and its eligibility for reward. From the date of its initiation (in 2004) Mozilla bug bounty program has paid out over 1.6 million dollars.

2.2 Platform Managed Vulnerability Discovery

This model is gaining increasing popularity. In this model, the platform providers (e.g., Bugcrowd, Cobalt, Synack, Hackerone and Bounty Factory) are responsible for managing the community of SecPros to perform the task: providing reputation policies to motivate SecPros (e.g., through gamification, badges), providing SecPros selection strategies and communicating with them, providing guidance to both the organization and SecPros, and managing rewards. Some platforms provide the choice of launching a program for a certain period of time (e.g., before launching a new software product), or keep the program running for an unspecified period.

An example of this model is Bugcrowd, which is a platform that was founded in 2012 and currently maintains a community of around 22,000 SecPros. Bugcrowd ranks SecPros according to their performance, based on indicators such as number of discovered vulnerabilities, quality of submitted reports and impact of discovered vulnerabilities. The platform provides support for handling vulnerability report submissions (e.g., filtering ineligible reports, such as duplicate and out-of-scope vulnerabilities reports). Bugcrowd also offers organizations the option to either run an on-going program (e.g., Heroku program), or a time-boxed program (e.g., Aruba Networks program), and to define the criteria of selecting the SecPros to be invited to participate.

2.3 Cyber Security Contests

In this model, software providers, platforms or security concerned organizations submit tasks for the discovery of vulnerabilities in a contest which is usually bound over a short period of time and a certain location. SecPros submit discovered vulnerabilities to

[3] www.mozilla.org/en-US/security/bug-bounty.

the organizers of the contest who will be responsible to assess submissions for validity and to determine the winner of the contest (e.g., according to severity of discovered vulnerabilities).

Pwn2Own is one of the most famous cyber security contests. It usually runs annually in conjunction with the CanSecWest security conference. SecPros eligible to participate must be registered and available on-site at the time of the contest. SecPros need to exploit a series of undisclosed vulnerabilities to compromise a system (e.g., computer or mobile phone). SecPros, participating in the contest, get quite the media cove rage and high rewards (e.g., total prizes for Pwn2Own2017 were around 800,000 dollars).

Table 1. Dimensions for characterizing crowdsourcing vulnerability discovery

Dimension	Brief Description	Possibilities
Crowd Size	Number of SecPros to participate in the task	Small; Medium; Large
Incentives	Motivations for SecPros to engage in the task	Monetary; fame and glory; passion
Duration of the Task	The amount of time the task would be running for	time-boxed; open: hybrid
Task Context	The information provided to SecPros to perform the task	None; little; medium; extensive
Task Management	The party responsible for managing the task (e.g., communicating with the crowd, and payments)	Organizations; platforms; shared between organizations and platforms.
Selection of SecPros	The selection of SecPros to participate in the task	SecPros self-select; organizations select according to certain criterion (e.g., credentials, experience, and reputation).
Information protection	The protection against exposure of sensitive information (e.g., customer information, PII), or Intellectual property (e.g., source code, unreleased software)	Manual methods; encrypting or masking; proxy access; staging environment; privacy policies; terms and conditions, limit the access of SecPros to public information; vetting SecPros; splitting the task to small modules
Legal Terms	The form of legal contract between the SecPros and the organization	No explicit legal terms relying on Anti-hacking laws, explicitly dictating weaving of legal pursuit when SecPros commit to terms and conditions, authorizing SecPros to discover vulnerabilities bonded by platforms term and conditions and the organization terms and conditions

3 Dimensions of Crowdsourcing Vulnerability Discovery

To compare, contrast and appreciate the underlying design of each of the models, we reviewed literature and the current implementations of crowdsourcing vulnerability discovery and identified eight dimensions that characterize these models (as shown in Table 1). We build upon other models that employ the use of crowdsourcing more generally [4, 6]. The identified dimensions can be used to describe not only the current models for crowdsourcing vulnerability discovery but also possible variations. Examples for each model are illustrated in Table 2.

3.1 Crowd Size

Greater number of participating SecPros means a higher probability that vulnerabilities will be discovered and reported [2, 3]. The burden to manage the crowd and verify the legitimacy of submitted vulnerability reports also increases [7]. In the direct model, the potential number of SecPros participating is large since the task is open for everyone. While for the platform managed model, the potential number is medium to small, since the organization controls the number of SecPros invited to participate and is limited by the size of the community maintained by the platform. On the other hand, in the contest model the potential number of SecPros are small since it is restricted by the capacity of the venue where the cyber security contest is held.

3.2 Incentives

It has been observed that the crowd is motivated by money, fame and glory, and passion [6, 8, 9]. In the direct vulnerability discovery model, a combination of money, fame and glory and passion is relied upon. SecPros, in addition to acquiring cash rewards, usually are mentioned in the hall of fame or gratitude page (e.g., Facebook white hat program). On the other hand, for the platform managed model, money and passion play an important factor in motivating SecPros. Additionally, platforms tend to rely on gamifications and ranking SecPros according to their contributions to increase the level of engagement (e.g., Hackerone leader board). For the cyber security contest model, the most important incentives are money, fame and glory since such contests often attract large attention from the media compared to other models.

3.3 Duration of the Task

In the direct model, the task duration is not time-boxed and hence SecPros may submit vulnerability reports at any time (ideally as soon as a vulnerability is discovered before other SecPros). In the platform managed model, the organization has the choice of running a time-boxed or open-ended program. Whereas, in the contest model the task is time-boxed which may indicate that it is the fastest form, but sometimes SecPros lean toward vulnerability hoarding (i.e., discovering vulnerabilities beforehand and not disclosing them until the time of the contest). Google stated that one of the reasons for terminating the Pwnium contest was to "remove the incentives for bug hoarding"[4]. On the other hand, it has been discovered that running the task for longer time give SecPros more time to discover vulnerabilities and potentially increase the number of vulnerabilities submitted [3].

3.4 Task Context

To perform vulnerability discovery tasks, SecPros may require information to help them on their quest (e.g., test dataset, instructions how to bypass a security mechanism, or source code of the software). Information about the software and development environment has been proven to be important to help SecPros to perform the task [9].

[4] blog.chromium.org/2015/02/pwnium-v-never-ending-pwnium.html.

In the direct model, the organization shares publicly available information. In the platform managed model, the organization is more comfortable sharing information (e.g., credentials to bypass web proxy, test accounts, access to staging environment), especially if the vulnerability discovery program is private (i.e., could be visible only to invited SecPros). In the contest model, little context (e.g., the software for testing and general rules) is provided in the beginning (as the task information is available publicly), but further information could be supplied on-demand to SecPros at the contest venue (e.g., access to the system through the network, entry points that can be used to simulate the attack). Task Management.

3.5 Task Management

Managing tasks includes managing vulnerability description reports (e.g., identifying out of scope reports, identifying duplicates, storing reports securely), communicating with SecPros (e.g., requesting clarification, requesting proof of concept), managing rewards (e.g., payments to SecPros, managing the hall of fame and gratitude pages). In the direct model, the organization is responsible for managing the task. In the platform managed model, the task is managed by the platform. In the contest model, the task could be managed by the organization, the platform or combination of both.

3.6 Selecting Security Professionals

Different vulnerability discovery tasks require that SecPros have different sets of skills, knowledge, and expertise (e.g., Web application vulnerabilities require knowledge about the software itself, networking protocols, Web frameworks, and the types of Web vulnerabilities). Hence, it is important to select qualified SecPros. Previous research identified the indicators to the expertise of SecPros participating in the crowdsourced task of vulnerability discovery [10]. Also, it has been discovered that selecting SecPros with high skill diversity help discover more diverse vulnerability types [8].

In the direct model, the SecPros self-select themselves, although organizations may sometimes impose entry conditions (e.g., Mozilla bug bounty guidelines require that no employee of the Mozilla foundation or affiliated service provider is eligible to participate). Whereas, in the platform managed model, the platform maintains SecPros' profiles including details provided by the SecPros themselves (e.g., certification, public professional profiles) along with their ongoing platform measured performance (e.g., number of vulnerabilities discovered, ranking against other SecPros). Organizations may invite SecPros within the community according to certain criterion (e.g., allow only the top 100 ranked SecPros, allow only CISSP certified SecPros). In the contest model, although SecPros self-select themselves to participate in the task, entry conditions may be imposed (e.g., traveling to the contest location).

3.7 Information Protection

Major privacy concerns arise when software providers are required to share data that is needed for vulnerability discovery. Even when providing anonymized (or synthetic) data, it is still possible that sensitive data be leaked by mistake or inferred via aggregation and correlation analysis. Other privacy threats may arise when the system has a vulnerability that would grant access to private information (e.g., a software glitch in the bicycle-sharing program Citi-Bike leaked customer details by accident, which led to a breach of privacy policies). Vulnerability description reports need to be protected as well, since their disclosure before discovered vulnerabilities are repaired could be damaging to organizations.

In the direct model, organizations rely on special controls to protect sensitive data from malicious attackers (e.g., automated tools to monitor access and report to internal staff for response and encrypting or masking sensitive data). In the platform managed model, since tasks are released to members of a closed community, all members of the community must consent to the terms and conditions imposed by the platform before participating in tasks. Moreover, since the number of invited SecPros is known, the organizations can provide proxy access to the task environment and monitor all the network traffic when SecPros are accessing the system in order to block any suspicious activity. In addition, a staging environment could be used for vulnerability testing (e.g., Movember bug bounty program run with Bugcrowd allowed SecPros to access a staging environment for vulnerability discovery). Similarly, in the contest model a prearranged environment is setup and SecPros agree to the terms and conditions before participating in tasks.

Intellectual Property leakage could result in the loss of competitive advantage, which may in turn result in financial losses (e.g., leakage of IP cost Sony millions of dollars in the latest Sony security breach incident). An organization may share sensitive intellectual property (e.g., source code for review, or access to an unreleased application) with SecPros. To protect IP, organizations in the direct model, limits access to public and restricted information to avoid potential risks, albeit this comes at a cost of reduced capabilities of the SecPros in performing vulnerability discovery. On the other hand, in the platform managed model the platform may choose to only disclose to SecPros with verified identities and backgrounds. In addition, SecPros are bound to sign nondisclosure agreements (NDAs). Similarly, in the contest model, all SecPros need to register to participate in the contest (i.e., verifying their identities) and sign NDAs before participation. Another possible solution to mitigate IP leakage is to split the task into micro tasks (e.g., modules of the system to be tested for vulnerabilities). By having access to only small chunks of code or modules of the software, SecPros will not be able to deduce the overall functionality of the system. In a similar context, LaToza et al. suggested a methodology to decompose programming work into micro tasks and found that the benefit of decomposing the task outweigh the overhead [11].

Table 2. Examples of crowdsourcing vulnerability discovery models

Dimension	Direct (Mozilla)	Platform Managed (Bugcrowd-Movember)	Contest (Pwn2Own)
Crowd Size	Large	Medium	Small
Incentives	Money; Passion; Fame; Glory	Money; Passion	Money; Fame; Glory
Duration of the task	Open	Time Boxed	Time Boxed
Task Management	Organizations	Platform	Organizations and Platform
Selecting SecPros	SecPros Self-select	The organization Select from a Closed Community	SecPros Self-select
Information protection	Rely on Internal Staff and Methods and Sharing only Public data	Staging Environment	Staging Environment
Legal Terms	No legal Authorization and only binding SecPros with Anti-hacking laws	Platforms T&C and Organization T&C weaving legal liability	Authorizing the SecPros Directly

3.8 Legal Terms

Relying on the crowd to discover vulnerabilities is governed by legal terms that represents a contract between SecPros and organizations. Missing detailed legal terms would cause inconvenience for both parties and it especially puts the SecPros at greater risks if the task is governed by the anti-hacking laws only [12]. In the direct model, organizations explicitly state legal terms as a binding contract (e.g., Microsoft have detailed legal terms and conditions stating what are the SecPros authorized to do, the obligations of the SecPro and what are the conditions in which Microsoft would take legal actions). In the platform managed programs, the platform has legal binding terms in addition to the legal terms the organization has (e.g., Hackerone has their own legal terms and conditions and the organization running the program can add their own legal terms). In the contest model, the organization and the SecPro will be under more specific and direct legal contract (e.g., SecPros to sign agreements to be able to participate).

4 Discussion and Future Research Directions

Reflecting on the above analysis, we identify a range of possible future research directions:

4.1 Improving the Quality of Vulnerability Tasks Descriptions and Reports

Ensuring the clarity and completeness of the task description is important to allow SecPros to perform the task properly [13], platforms may employ machine learning

models to assess task description quality regarding clarity, specificity and completeness. This may involve textual analytics and machine learning algorithms that gauge the effectiveness of crowd task postings. Additionally, the verification and repair of vulnerabilities would be delayed if poorly written reports are submitted, as it would take more time to identify and understand the problems. This is exemplified in an incident where a SecPro exploited a vulnerability in Facebook to prove its legitimacy after being rejected due to quality issues[5].

It would also be interesting to investigate natural language processing (NLP) techniques to review the submitted vulnerability reports, and to possibly provide warnings or suggestions to improve the quality of the report prior to submission. One aspect worth investigating is whether the vulnerability described in the submitted report is within the scope of the task and notifying the SecPro before the report is submitted. This will minimize the number of invalid reports the organization may receive. Another interesting approach would be to rely on the crowd to provide feedback and enhance quality, or even to verify the legitimacy of the vulnerability reports. Allowing SecPros to assess and comment on the reports could help improve the quality of the vulnerability reports. In a similar context, Top-Coder relies on the crowd to provide code review for submitted software[6]. In the same context but by different approach, Su and Pan proposed a platform where administrators can collaborate on vulnerability report verification [14].

4.2 Protecting Against Intellectual Property Leakage

It is also important to investigate task decomposition techniques in order to ensure that SecPros will not have full access to the source code, or full view of the functionality of the software. However, an important challenge is to identify what types of vulnerabilities could be discovered in isolation and which types require complete view of the source code. As an example, buffer overflow and over-bound memory reads could be considered as vulnerabilities that could be discovered in isolation while business logic vulnerabilities may not be that straightforward. Investigating the different types of vulnerabilities, and how their discovery could be facilitated is an important research direction.

An additional interesting research direction would be to investigate what type of vulnerability discovery model would be most suitable for the type of software being tested. As an example, the direct model would be more suitable for open-source software since the IP leakage concerns would be minimized.

4.3 Crowdsourcing Vulnerability Discovery Quality Analytics

Although a wealth of information about SecPros and organizations are available (e.g., profiles, achievement portfolios, work history), most remains largely unharnessed.

A possible research direction is to model and capture the SecPros' behavioral patterns as a summary in terms of both aggregated work metrics (e.g., reputation scores, average time spent on tasks, number of unaccepted vulnerability reports) and semantically

[5] techcrunch.com/2013/08/18/security-researcher-hacks-mark-zuckerbergs-wall-to-prove-his-exploit-works.

[6] www.topcoder.com.

meaningful behavior categories (e.g., honest, deceiver or colluding). Recent empirical research has discovered various categories of crowd workers (e.g., ineligible, fast deceivers, smart deceivers, gold start preys) [15].

In order to predict the quality of the vulnerability report, it is important to investigate non-intrusive and statistical analysis techniques over the SecPros' activity history for given task types. A key issue that is not considered in existing techniques is uncertainty related to predicting performance of new SecPros (i.e., cold start case). Relying on indicators extracted from various sources (e.g., activities in SecPro communities like BlackHat[7], contribution to open-source projects in GitHub), it would be possible to "predict" the quality of submitted vulnerability discovery reports and detect the malicious behavior of the SecPros.

Using dynamic access control based on the allocation of "roles" is an interesting direction to investigate. For instance, a SecPro who is not known to a vulnerability discovery platform is allocated to the "untrusted" role. Their trust level can increase as the platform collects more information about them (e.g., task performance or reputation in external platforms). Moreover, the trust level would also increase as the SecPros improve their reputation score (e.g., discovering new vulnerabilities and disclosing them responsibly, or acquiring some achievement on other platforms). On the other hand, if SecPros did not adhere to the guidelines or best practices, then their trust would decrease.

Another interesting research direction is to investigate how to combine crowdsourcing with machine automation to improve vulnerability discovery. Votipka et al. characterized the process of vulnerability discovery [9]. A possible direction is investigate which phases of the process will benefit from automation and which phases will be benefit from human computation, as well investigation of hybrid (i.e., combing human and machine computation [16]). As an example, tasks like information gathering and attack surface detection can take advantage from automation. This can save time and effort for SecPros who can focus on other tasks like vulnerability recognition.

5 Conclusion

There are several models for crowdsourcing vulnerability discovery and these models are characterized by different dimensions. Understanding the models and dimensions for crowdsourcing vulnerability discovery helps practitioners make informed decisions when adopting this approach. It also paves the way for researchers to conduct a deeper investigation of the area if crowdsourcing is to have the same kind of impact in vulnerability discovery that it has in other fields.

References

1. Finifter, M., Akhawe, D., Wagner, D.: An empirical study of vulnerability rewards programs. In: Proceedings of the 22Nd USENIX Conference on Security, pp. 273–288 (2013)
2. Maillart, T., Zhao, M., Grossklags, J., Chuang, J.: Given enough eyeballs, all bugs are shallow? Revisiting Eric Raymond with bug bounty programs. J. Cybersecur. **3**, 81–90 (2017)

[7] www.blackhat.com.

3. Zhao, M., Grossklags, J., Liu, P.: An empirical study of web vulnerability discovery ecosystems. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security - CCS 2015, pp. 1105–1117 (2015)
4. LaToza, T., van der Hoek, A.: Crowdsourcing in software engineering: models, motivations, and challenges. IEEE Softw. **33**(1), 74–80 (2016)
5. Al-Banna, M., Benatallah, B., Schlagwein, D., Bertino, E., Barukh, M.: Friendly hackers to the rescue: how organizations perceive crowdsourced vulnerability discovery. In: Pacific Asia Conference on Information Systems (PACIS) (2018)
6. Malone, T.W., Laubacher, R., Dellarocas, C.: The collective intelligence genome. IEEE Eng. Manag. Rev. **38**(3), 38 (2010)
7. Laszka, A., Zhao, M., Grossklags, J.: Banishing Misaligned Incentives for Validating Reports in Bug-Bounty Platforms, pp. 161–178. Springer, Cham (2016)
8. Zhao, M., Grossklags, J., Chen, K.: An exploratory study of white hat behaviors in a web vulnerability disclosure program. In: Proceedings of the 2014 ACM Workshop on Security Information Workers - SIW 2014, pp. 51–58 (2014)
9. Votipka, D., Stevens, R., Redmiles, E., Hu, J., Mazurek, M.: Hackers vs. testers: a comparison of software vulnerability discovery processes. In: 2018 IEEE Symposium on Security and Privacy (SP), pp. 374–391 (2018)
10. Al-Banna, M., Benatallah, B., Barukh, M.C.: Software security professionals: expertise indicators. In: 2016 IEEE 2nd International Conference on Collaboration and Internet Computing (CIC), pp. 139–148 (2016)
11. LaToza, T.D., Ben Towne, W., Adriano, C.M., van der Hoek, A.: Microtask programming. In: Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology - UIST 2014, pp. 43–54 (2014)
12. Gamero-Garrido, A., Savage, S., Levchenko, K., Snoeren, A.C.: Quantifying the pressure of legal risks on third-party vulnerability research. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security - CCS 2017, pp. 1501–1513 (2017)
13. Zhao, M., Laszka, A., Grossklags, J.: Devising effective policies for bug-bounty platforms and security vulnerability discovery. J. Inf. Policy **7**, 372 (2017)
14. Su, H.-J., Pan, J.-Y.: Crowdsourcing platform for collaboration management in vulnerability verification. In: 2016 18th Asia-Pacific Network Operations and Management Symposium (APNOMS), pp. 1–4 (2016)
15. Gadiraju, U., Kawase, R., Dietze, S.: Understanding malicious behavior in crowdsourcing platforms: the case of online surveys. In: 33rd Annual ACM Conference on Human Factors in Computing Systems, pp. 1631–1640 (2015)
16. Krivosheev, E., Casati, F., Baez, M., Benatallah, B.: Combining crowd and machines for multi-predicate item screening. Proc. ACM Hum.-Comput. Interact. **2**(CSCW), 1–18 (2018)