Using Amazon's Mechanical Turk for User Studies

Eight Things You Need to Know

Lucas Layman
Fraunhofer Center for Experimental Software Engineering
College Park, MD, USA
llayman@fc-md.umd.edu

Gunnar Sigurðsson
School of Computer Science, Reykjavik University
Reykjavik, Iceland
gunnar11@ru.is

***Abstract*—Amazon's Mechanical Turk is a crowdsourcing technology that enables requesters to create tasks to be completed by human agents in exchange for compensation. Researchers in computer science have successfully used this service to quickly reach large numbers of subjects for a relatively low cost. However, the Mechanical Turk's model and policies introduce several experimental limitations and threats that must be controlled. In this short paper, we describe limitations imposed using Amazon's Mechanical Turk during an experiment on cyber-attack investigation techniques. While the experiment was successful, we were forced to change our experimental design and had to recover from some costly mistakes. The goal of this short paper is to identify these limitations and pitfalls and provide eight considerations for experimental design so that other researchers can maximize the benefits of using the Mechanical Turk as a research platform.**

Keywords—mechanical turk; experimentation; user studies; amazon; study design

I. INTRODUCTION

Amazon's Mechanical Turk is a crowdsourcing service where *requesters* create online tasks for *workers* to perform in exchange for monetary compensation. The service is intended to hire individuals to complete short tasks requiring human intelligence without the legalities of actual employment. Researchers have shown that the Mechanical Turk can be used to perform a variety of robust studies in both the social (e.g., [1], [2]) and computer sciences (e.g., [3], [4]). User studies in software engineering and human-computer interaction are often limited by the number of qualified participants accessible by the researchers. The Mechanical Turk can help alleviate this problem by providing access to a worldwide pool of potential research subjects at a relatively low cost.

The Mechanical Turk can be and has been used for both simple research tasks and complex user studies. Since interaction in the Mechanical Turk is asynchronous, the scope of studies is limited to those that can be conducted online. Further, the mechanics of the service decrease the experimental control the researcher has over the subjects. Nonetheless, the ability to access a huge number of subjects makes the Mechanical Turk a high-potential research tool. An overview of the benefits and challenges associated with using the service as a research platform for empirical studies can be found in [5].

We used the Mechanical Turk to conduct a user study to understand which information in webserver log files is most useful when investigating potential cyber-attacks. We encountered a few challenges in using the Mechanical Turk service that impacted both the execution and validity of our experiment. In the end, we overcame these challenges at the expense of time and money and validated our findings from a previous controlled experiment.

The goal of this short paper is to identify some limitations and pitfalls that may be encountered when conducting online user studies with the Mechanical Turk so that other researchers can maximize the benefits of using this service as a research platform. We provide our own study as an example. We expand upon the initial evaluation of the Mechanical Turk for empirical studies in [5] and [4] and identify eight considerations for experimental design and discuss how we overcame the experimental limitations and challenges.

II. WHAT IS AMAZON'S MECHANICAL TURK?

From the website (www.mturk.com), the Mechanical Turk "give[s] businesses and developers access to an on-demand, scalable workforce. Workers select from thousands of tasks and work whenever it's convenient". The basic scenario of the Mechanical Turk is that a *requester* creates a *human intelligence task (HIT)* that is completed by a *worker*, who is then compensated. For example, a market researcher creates a task where several different logos are presented on the screen and a worker selects her favorite. In exchange for completing the task, the requester pays the worker $0.05. The requester can define the number of workers who can complete the HIT, and specify how many times and individual worker can complete a task. There are many features and work flows in the Mechanical Turk that we do not cover here.

Requesters currently have two options for hosting the HITs. First, HITs can be created and hosted on Amazon's Mechanical Turk site. This limits the HITs to single page HTML forms that support a variety of functions, e.g., image display, embedded JavaScript, HTML5, and Flash. While robust, this option does not easily support multi-step tasks, e.g., first provide your personal information, then complete Task 1, then Task 2, etc. The second option is for the requester to host the webpage on his own webserver, which is embedded in an HTML iframe (inline frame) on the Mechanical Turk site. Numerous parameters can be configured for a HIT, such as a time limit, the number of times an individual worker can complete the HIT, and the total number of copies of the HIT available to the workforce.

978-0-7695-5056-5/13 $26.00 © 2013 IEEE
DOI 10.1109/ESEM.2013.42

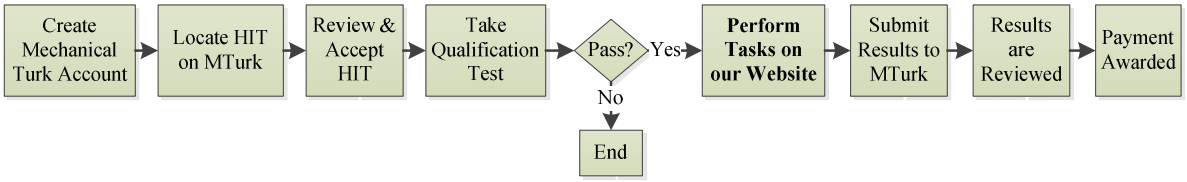

Fig. 1. Example workflow of a worker completing our HIT on the Mechanical Turk

Requesters can define *qualification types* that must be met by potential workers before they can begin a HIT. Qualification types include restricting workers to a particular country, requiring a specified HIT completion percentage (i.e., to avoid workers who abandon tasks), or requiring workers to pass a custom-built "qualification test" defined by the requester. Requesters can also *verify HIT results* to ensure that the workers completed the task satisfactorily before being awarded compensation. All functions for creating HITs, defining qualification types, reviewing results, and more can be accessed through either the Mechanical Turk web interface or a rich set of open source APIs available in several languages.

Workers use the Mechanical Turk website to search for interesting tasks. Workers can look for tasks from a known requester, search by keyword, the estimated time required to complete the task, or by the total compensation. There are currently over 193,000 HITs available on the Mechanical Turk. Most HITs take less than one minute to complete and offer between $0.01 and $0.10 USD in compensation. Compensation is handled automatically through Amazon's user services, so no communication is required between requester and worker. Amazon verifies worker's real identities to ensure that an individual can only have one worker account to prevent fraudulent work and compensation using multiple accounts. An example workflow from the perspective of a Mechanical Turk worker is shown in Figure 1.

III. CYBER ATTACK STUDY OVERVIEW

In previous research, we conducted a controlled experiment with 14 subjects (most of whom were students) to identify the data in webserver log files that is most useful for investigating potential cyber-attacks and to understand how these data are used. In the controlled experiment, we recruited subjects from a convenience sample and collected demographic information. We presented the subjects with six webserver log files on a laptop, one at a time, and asked them to determine whether the activity in the log file was malicious or not and to explain their reasoning. Finally, the subjects were asked several debriefing questions. The experimental sessions were time-limited, audio recorded, and screen-captured. The sessions were conducted with the researcher in the same room as the subject, and all subjects were compensated. In our analysis, we examined which data fields in the log files subjects interacted with, performed a qualitative analysis of their debriefing responses, and examined the relationship of their maliciousness-assessment analyses and their demographic information.

After this initial controlled experiment, we wanted to validate our findings by expanding our subject pool to include more subjects, particularly professionals. We determined that the Mechanical Turk could help us to meet this objective. To use the Mechanical Turk service, our experiment had to be converted to a web-based study that could be completed asynchronously without direct researcher supervision. Also, we would not have the richness of the screen or audio recordings, and we assumed that we had to offer compensation commensurate with professional pay in order to attract professionals with expertise in security. The transition to the Mechanical Turk introduced a number of potential threats to validity that we did not need to explicitly consider during the original controlled experiment:

- Qualifications of subjects – how do we ensure the subjects from the workforce are from the desired target population?
- Data validation – how do we ensure that subjects will not game the system to receive compensation? How do we recognize when data is invalid?
- Procedure adherence – how can we enforce that subjects complete the experimental tasks in order and that time limits for individual log files analysis are enforced?
- Independence of observations – how do we prevent subjects from sharing answers to the analysis questions?

We implemented the controlled experiment as a series of webpages implemented in ASP.NET MVC 4 on a Fraunhofer webserver. A summary of the experimental procedure seen by Mechanical Turk workers is shown in Figure 2. The questions asked by the researchers during the original experiment were converted to HTML forms. The subjects' responses to questions, log file analysis results, and time spent on each web page were stored in a database on the Fraunhofer webserver.

We also defined a qualification type on the Mechanical Turk that was used to screen the subjects. The qualification type was a web form containing 12 multiple choice questions on web server knowledge. In order to be eligible to start the experimental task, the subjects had to achieve a high score on the qualification test and complete the test within 2.5 minutes. Qualified workers could then accept the HIT and the experimental webpages were shown in an HTML iframe embedded within the Mechanical Turk site. Workers completed the experimental tasks and the results were submitted to the Mechanical Turk. Every night, a researcher would review the submissions and approve compensation if the data was deemed to be valid.

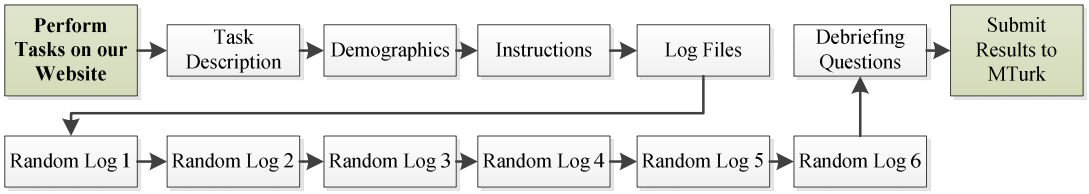

Fig. 2. Experimental webpages and flow on the custom task website. Each grey block is an individual web page.

IV. EIGHT THINGS YOU SHOULD KNOW

Converting our controlled user study to an online study compatible with the Mechanical Turk introduced several threats to validity: some anticipated and some unexpected. We adjusted our experimental procedures to account for these threats, but at the expense of time and money. While some of the lessons learned below are particular to our experimental structure, our goal is to inform researchers of these risks to promote a more robust study design. A total of 53 subjects participated in the Mechanical Turk study over a two-week period, two of whom were disqualified for attempting to game the system. Of the 51 remaining subjects, 39 were professional IT administrators, software engineers, or security experts. The results of both studies can be found in [6].

A. *Subject screening capabilities are flexible and sufficient*

One of our biggest concerns was the threat of workers who would manipulate our HIT to receive compensation with minimal effort (see [7]). Using the Mechanical Turk's built-in screening mechanisms, we designed a qualification test that a knowledgeable person could easily pass, whereas someone looking up answers would not complete the test on time. Our questions focused on negatives that would be harder to search, e.g., the answer to "which of the following is not a protocol" involves searching for each of the four possible answers. Researchers can also manually qualify workers for a HIT using the worker's unique ID. This method may be useful if more sophisticated offline subject screening mechanisms are used, such as a phone interview or custom survey. Implementing the qualification on the Mechanical Turk limits the amount of involvement required on the part of the researcher. Ultimately, we were satisfied with the qualification services, and do not believe the subject screening threatened our validity. The accuracy of the Mechanical Turk subjects in identifying malicious log files (75.1%) was approximately equivalent to the accuracy of the controlled study subjects (74.8%).

B. *Make data validation possible;reject submissions carefully*

Researchers must be able to validate workers' submitted data quickly. The Mechanical Turk can automatically approve and compensate a worker for their submission, but many researchers will want to validate the worker's data prior to compensation. In our study, we tracked how much time a subject spent on each log file to identify subjects who merely clicked through to receive compensation. We rejected two submissions that we believed fell into this category. It is essential to have a defensible methodology for rejecting submissions, as workers can challenge the decision with Amazon. Furthermore, rejecting a worker's submission on Mechanical Turk will both deny compensation and negatively impact their reputation, which affects their ability to participate in future tasks. Thus, rejecting a worker's submission must be done with care.

C. *Achieve procedure adherence through robustness, not by implementing technical controls*

In our study, it was important for the subjects to complete the demographics, the six log file analyses, and the debriefing questions in a specific order. Also, we did not want subjects to go back to change their answers. In the controlled experiment, the researcher in the room shepherded the subject through these steps. In the online study, we first attempted to restrict the subjects' abilities to navigate on the website. We implemented JavaScript methods for detecting a press of the back button and checks on each web page to see if it had been previously visited. Due to the non-deterministic nature of web browsing, the logic for trying to control navigation in this manner was both convoluted and limited us to subjects who had JavaScript enabled. Many security-conscious users disable JavaScript in web browsers as a security precaution.

To address this challenge, we recommend making website *tolerant* to aberrant navigations. We implemented server-side logic (as opposed to browser-based logic) that recorded which web pages subject had visited and verified the form data. If a subject browsed to a webpage she previously visited either by hitting the back button or navigating directly, she would be taken to the first page in the experimental order she had *not* previously visited. The subjects were identified by a unique ID provided by the Mechanical Turk in the HTTP header. This method was simpler, more robust, and easier to test.

D. *Subject socialization is not an automatic threat to independence of observations*

One of our greatest concerns was that Mechanical Turk workers would share answers to the analyses and qualification tests. Several workers indicated that they heard about our HIT from cloud worker forums. We visited these forums and observed no signs of answer sharing. In fact, these forums no doubt assisted in recruiting subjects. Further, Mechanical Turk workers earn a reputation score (tracked by Amazon) based on the number of HITs they abandon or are rejected by the requester. A low score can make workers ineligible for HITs, prompting workers to safeguard their reputations. We suspect that our high compensation and the threat to their reputations incentivized workers to complete the task with integrity lest we exclude them from future studies. Ultimately, the cloud worker forums were a boon to our study.

E. Be aware of asynchronous task engagement by workers

In the controlled experiment, subjects were limited to 10 minutes to analyze each log file to minimize fatigue and to accommodate their schedules. The Mechanical Turk can limit the time allowed to complete a HIT, but finer-grained controls are unavailable. In piloting the Mechanical Turk HIT, a number of users encountered errors due to the timing control mechanism, which prevented them from generating any usable data. We removed the timing control from our task site, and set the total time that could be spent on the HIT to 90 minutes. Our website provided warnings when there were 10 minutes and five minutes left on the study. However, the lack of timing control introduced another threat to our data because the subjects could disengage from the research task to attend to other matters. This occurred at least once. We excluded the outlier from our data analyses and were forced to acknowledge that any time measurements had an unknown error margin.

F. Compensation does not need to be exorbitant

We assumed that IT professionals in the US would make around $30/hour, and thus we offered $30 for completing our HIT. Our HIT was one of the highest paying of all those offered on the Mechanical Turk. The effect was three-fold: 1) our HIT stood out in HIT searches; 2) workers were eager to qualify for the HIT (as evidenced by emails sent to us by workers who failed qualification); and 3) the HIT was circulated among social circles of Mechanical Turk workers. After discovering a bug, we recreated our HIT and lowered the compensation to $15 with no apparent effect on participation. We asked our participants if they had any feedback for our HIT. Many of them commented that they enjoyed it, specifically mentioning that they enjoyed working on a HIT that compensated so well *and* made use of their technical skills. We assume that many Mechanical Turk workers work during downtime at their current jobs or after hours, and that the quick, mindless nature of many hits (e.g., image tagging) becomes tiresome when working for $0.02 per task. Thus, the bar for compensation seems quite low if the task is interesting. In total, we paid $1039.50 for ~60 participants, including partial compensation for people due to errors on our task site.

G. Managing post-release bugs on a complex task site requires a maintenance infrastructure

To protect both the requesters and the workers, the Mechanical Turk implements a strict process for creating and deleting HITs, managing qualifications, rejecting results, and paying workers. This process has significant ramifications for study execution. For example, assume that 50 workers pass a qualification test for your HIT. Later, you discover this test has an error. The ramifications are: 1) you must discard the data from erroneously qualified workers that completed the HIT but still must compensate them; 2) you have to take your experiment offline when it may have gathered social momentum; and 3) you have to recreate the qualification test, which may be time consuming.

To counter this problem, we used the Mechanical Turk Java API to create a small application for creating and disabling our HIT, creating qualification types, and reviewing the HIT status. This was essential for quickly making changes to HITs and qualification tests, and we recommend that all researchers take advantage of the programming APIs. Furthermore, we *strongly* recommend a thorough modeling of the steps required to adjust your Mechanical Turk task in the event of (inevitable) errors in your HIT or qualification tests.

H. Do not prepay for many HIT submissions

To publish a HIT on the Mechanical Turk, a requester must deposit sufficient funds into their Amazon account to cover the cost of the specified maximum number of workers completing the HIT plus a 10% Amazon fee. For a $30 HIT and 10 participants, the requester must have $330 in his account. Once a HIT is active, you cannot reduce the number of total participants it will accept. Thus, to reduce the number of participants (or the amount of compensation), you must completely delete the HIT and create a new HIT, which means ensuring the same subjects do not participate in both HITs. Thus, we recommend that researchers only pay for small batches of a HIT, and that they monitor the progress of the HITs and increase the available tasks incrementally.

V. Summary

This short paper describes several limitations and threats to validity encountered while replicating a controlled study using Amazon's Mechanical Turk. While successful, several changes had to be made to the study to account for unexpected difficulties when using the Mechanical Turk. We hope that our recommendations will be useful to other researchers interested in using the Mechanical Turk to gain access to a wider pool of potential research subjects for user study research.

References

[1] G. Paolacci, J. Chandler, and P. G. Ipeirotis, "Running Experiments on Amazon Mechanical Turk," *Judgement and Decision Making*, vol. 5, no. 5, pp. 411–419, Jun. 2010.

[2] M. Buhrmester, T. Kwang, and S. D. Gosling, "Amazon's Mechanical Turk: A New Source of Inexpensive, Yet High-Quality, Data?," *Perspectives on Psychological Science*, vol. 6, no. 1, pp. 3–5, Feb. 2011.

[3] P. G. Ipeirotis, F. Provost, and J. Wang, "Quality management on Amazon Mechanical Turk," in *Proceedings of the ACM SIGKDD Workshop on Human Computation - HCOMP '10*, 2010, pp. 64–67.

[4] A. Kittur, E. H. Chi, and B. Suh, "Crowdsourcing user studies with Mechanical Turk," in *Proceeding of the twenty-sixth annual CHI conference on Human factors in computing systems - CHI '08*, 2008, pp. 453–456.

[5] K. T. Stolee and S. 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 - ESEM '10*, 2010, pp. 35:1–35:4.

[6] L. Layman, "Information and Processes Used when Investigating Web Server Log Files for Malicious Activity." TR 13-101 http://goo.gl/du73k) College Park, MD, 2013.

[7] J. S. Downs, M. B. Holbrook, S. Sheng, and L. F. Cranor, "Are your participants gaming the system?," in *Proceedings of the 28th international conference on Human factors in computing systems - CHI '10*, 2010, pp. 2399–2402.