Hindawi
Security and Communication Networks
Volume 2021, Article ID 8859618, 12 pages
https://doi.org/10.1155/2021/8859618

Research Article

Remote Attestation on Behavioral Traces for Crowd Quality Control Based on Trusted Platform Module

Donglai Fu [1,2] **and Yanhua Liu** [3]

[1]*Software School, North University of China, Taiyuan 030051, Shanxi, China*
[2]*Shanxi Province Military-Civilian Integration Software Engineering Technology Research Center, Taiyuan 030051, Shanxi, China*
[3]*Affiliated Hospital, North University of China, Taiyuan 030051, Shanxi, China*

Correspondence should be addressed to Donglai Fu; hhluci@163.com

Received 14 September 2020; Accepted 15 April 2021; Published 27 April 2021

Academic Editor: Leonardo Mostarda

Behavioral traces of workers have emerged as a new evidence to check the quality of their produced outputs in crowd computing. Whether the evidence is trustworthy or not is a key problem during the process. Challenges will be encountered in addressing this issue, because the evidence comes from unknown or adversarial workers. In this study, we proposed an alternative approach to ensure trustworthy evidence through a hardware-based remote attestation to bridge the gap. The integrity of the evidence was used as the trustworthy criterion. Trusted Platform Module (TPM) was considered the trusted anchor inspired by trusted computing to avoid unreliable or malicious workers. The module carefully recorded and stored many workers' behavioral traces in the storage measurement log (SML). Each item in the log was extended to a platform configuration register (PCR) by the occurrence sequence of each event. The PCR was a tamper-proof storage inside the TPM. The value of the PCR was also considered evidence together with the SML. The evidence was sent to the crowdsourcing platform with the TPM signature. The platform checked the integrity of the evidence by a series of operations, such as validating the signature and recomputing the SML hash. This process was designed as a remote attestation protocol. The effectiveness, efficiency, and security of the protocol were verified theoretically and through experiments based on the open dataset, WebCrowd25K, and custom dataset. Results show that the proposed method is an alternative solution for ensuring the integrity of behavioral traces.

1. Introduction

In China, crowdsourcing has rapidly progressed in various fields in the past years. Zhubajie (http://www.zbj.com) has established itself as a crowdsourcing leader with more than 22 million active workers. It covers a range of online and offline services, including tutoring and logo and product designs. Didi Chuxing (DiDi) (https://www.didiglobal.com) is another representative example of crowdsourcing platform. DiDi is China's leading mobile transportation platform that provides a full range of app-based transportation services, including taxi, express, premier, deluxe, bus, designated driving, enterprise solutions, bike sharing, e-bike sharing, automobile solutions, and food delivery, for many people. Tens of millions of drivers find flexible work opportunities on the DiDi platform. This platform provides more than 10 billion passenger trips annually. Crowdsourcing has become a fast, convenient, and cost-effective mode of research and production to obtain flexible and cheap resources. Many organizations flexibly outsource work, such as collaborative sensing [1, 2] and human-powered online security [3, 4], to a global pool or workers on a temporary basis. However, quality control remains a challenge in crowdsourcing systems, because the crowd is typically composed of people with unknown and extremely diverse abilities, skills, interests, personal objectives, and technological resources.

In crowdsourcing systems, tasks are posted to a crowdsourcing platform and are solved by a large group of workers with diverse characteristics. Amazon Mechanical Turk (MTurk) [5] is a successful crowdsourcing system that enables individuals or companies to harness collective intelligence from a global workforce to accomplish various tasks, such as human intelligence tasks (HITs). Employers (known as requesters) recruit employees (known as workers) to execute HITs, evaluate the outputs produced by workers, and reward them depending on the quality. The produced outputs may be directly and automatically checked by the crowdsourcing platform when requesters delegate responsibility for quality control. Rewards may be monetary or nonmonetary, such as gifts and reputation.

Quality control is important and challenging in crowdsourcing, because of the heterogeneous nature of workers [6, 7]. Output quality depends on the profiles and abilities of workers, description of tasks, incentives provided, processes implemented to detect malicious behavior or low-quality data, and the reputation and collaboration by the requester. Therefore, output quality is related to many dimensions. In the current study, only workers' dimension is considered. Although most workers perform tasks in good faith and are competent to deliver quality outputs, not all outputs are of high quality. Ensuring the quality of outputs remains challenging, and many algorithms based on workers have been proposed for quality control. Gold data are widely applied in this scenario. One approach is to adopt a qualification test, in which a worker is given questions where the answers are already known, to determine his or her performance. The famous crowd platform, CrowdFlower, uses this method to estimate the quality of workers [8]. Another approach is to mix gold tasks into tasks that are assigned to workers. Different from the former, workers do not know that this is the golden task, and they do not perform a test at their first arrival. The two approaches require the ground truth of a subset of tasks to be known in advance. Many problems are found in using these methods. How better and faster can we estimate the quality of workers when we use gold data? What is the improvement rate of quality estimation with the addition of many gold data? Generating high-quality gold data is expensive. These methods may be costly, because they are additional operations in crowdsourcing.

A recent tendency is by leveraging workers' behavioral traces to estimate the quality of outputs. The novel method attempts to capture the process that workers use to perform tasks by a series of events. This approach has advantages over the aforementioned methods. First, the monitoring program is almost invisible to the workers, because it runs in the background. Therefore, fraudsters who attempt to evade the anticheating checks will be foiled. Second, the time and cost for conducting experiments can be reduced because the monitoring will not introduce extra or redundant questions. Existing methods based on workers' behavior traces have mainly focused on what metrics should be collected and detection methods based on metrics. Limited researches have been conducted on trustworthy behavioral traces that are preconditions in making correct decisions.

In this study, we propose an alternative approach to ensure trustworthy behavioral traces with trusted computing for bridging the gap. In particular, integrity is considered the trust criterion. In the current study, these terms, integrity, trust, and authenticity have no difference. A Trusted Platform Module (TPM) embedded on the motherboard of a computer is considered the trusted anchor to avoid human factors. It faithfully records and reports workers' behavior. The behavioral traces are stored in a storage measurement log (SML). Each behavioral trace is extended to a tamper-proof register called platform configuration register (PCR), by its occurrence sequence. The PCR value is considered the evidence combined with the SML. When reporting the evidence, the TPM makes the signature using its private key to ensure the integrity of behavioral traces. The crowdsourcing platform can check the assurance through a series of operations, such as validating the signature and recomputing the SML hash. This process is designed as a secure remote attestation protocol (SRAP) to ensure that it is done on the basis of the expected behavior.

This study investigates a fundamental problem to ensure the integrity of the evidence in crowdsourcing quality control on the basis of workers' behavioral traces. The principal contributions of this work are summarized as follows:

(1) The current problem is formulated as a remote attestation protocol (RAP) on the basis of assumption, threat model, and attack types. The security of the protocol is defined on the basis of an adversary experiment.

(2) A specific SRAP, SRAP-I, is designed between a worker and a crowdsourcing platform. The protocol inventively utilizes one TPM binding a physical computer as the trust anchor to avoid attacks from malicious workers or malware. To the best of our knowledge, the current paper is the first to solve the issue on ensuring the integrity of behavioral traces of workers in crowdsourcing quality control.

(3) Evaluation is performed on a custom dataset and an open MTurk dataset of behavioral traces during relevance judging of search results, covering 106 unique workers and 3,984 HITs. The results show that SRAP-I is effective, efficient, and secure.

The remainder of this paper is organized as follows. Section 2 explores the related work. Section 3 formulates the current problem. Section 4 describes the specific SRAP, SRAP-I, in detail and presents the mathematical proof of its security. Section 5 discusses the ability to resist attacks. Section 6 conducts the experimental evaluation. Section 7 provides the conclusions.

2. Related Work

The combination of humans and computers to accomplish tasks that cannot be performed alone has attracted considerable attention from academic and industrial circles [9]. This idea dates back to the 1960s, with the publication of

"Man–Computer Symbiosis" by Likelier [10]. Tim Berners-Lee proposed the concept of a social machine in 2009 and regarded the cooperation between machines and humans as the next direction of web application development [11]. The term "crowdsourcing" was coined by Jeff Howe in 2006 [12]. MTurk is a pioneering crowdsourcing system that provides on-demand access to task forces for microtasks that are easy for humans but remain difficult for computers.

The outputs produced by crowd workers include many noises. This condition is because of many aspects. First, these workers might have different skill levels that are sometimes insufficient to complete the tasks. Second, they might have various and biased interests and incentives. Finally, malicious workers who intentionally provide incorrect answers are found. Many other factors, such as task description and data quality, affect the quality of outputs. Quality control has been widely investigated because of these features with the emergence of new technologies.

Substantial research efforts have been exerted to develop methods for detecting and correcting low-quality work to improve the overall quality of the resulting data. The state-of-the-art quality control can be categorized into three classes, namely, individual, group, and computation-based, in accordance with the literature [6]. The current study mainly focuses on fingerprinting, which is a computation-based method. This method captures behavioral traces from workers during task execution and uses them to predict the quality, errors, and likelihood of cheating. Rzeszotarski and Kittur first coined a method where the interactions of a crowd worker with a task interface, such as clicks, scrolls, and key presses, are logged and then correlated with his/her accuracy [13]. They demonstrated the effectiveness of the approach in predicting output quality. As an extension, the authors in [14] proposed a system called CrowdScape that supports the human evaluation of complex crowd work through interactive visualization and mixed initiative machine learning. Heymann and Molina presented a novel analytic tool called "Turkalytics" for human computation systems. Turkalytics processes and reports logging events from workers in real time and has been shown to scale to more than 100,000 logging events per day [15]. Kazai and Zitouni collected behavioral data from crowd workers and experts through search relevance judging. They then trained a classifier to detect poor quality work on the basis of behavioral data. They concluded that accuracy is approximately doubled in several tasks with the use of gold behavior data [16]. Dang et al. built a framework called MmmTurkey by leveraging the concept of tracking worker activity [17]. They collected and shared a new MTurk dataset of behavioral signals in judging the relevance of search results on the basis of their framework [18]. They concluded that behavioral data can be effectively used to predict work quality using their prediction model. Gadiraju et al. studied the relation between behavioral data and performance of workers in microtasks. They demonstrated that behavioral traces are effective in selecting workers using a novel model [19]. Mok et al. applied a method based on the behavioral data of workers in Quality of Experience to detect low-quality workers [20]. Although trustworthy behavioral traces are the preconditions for the abovementioned method, this point is not apparently stated. To the best of our knowledge, no direct research is reported on this issue.

To bridge the gap, this study first focuses on the issue and proposes an alternative solution inspired by trusted computing. Trusted computing aims to develop technologies that ensure users about the behavior of the software running on their devices. In particular, a device can be trusted when it consistently behaves in the expected manner for the intended purpose [21]. Although software-based trusted computing architecture with interesting results has been proposed [22], it can only be used in limited settings and cannot provide the same security guarantees as hardware-based architectures. An important part of trusted computing is to protect against attackers for gaining full control over the system; that is, any application and operating system (OS) can be exploited. Hardware-based architectures protect applications from a malicious OS. No software-only solution can provide these guarantees, because an attacker can continuously manipulate the software when the OS is untrusted. An attacker cannot modify hardware functionality when it is considered immutable. Therefore, a user's trust is claimed to be rooted in the hardware, making hardware-based architecture only considered in this study.

Hardware-based remote attestation mainly depends on a secure chip, TPM. It is a hardware cryptographic module consisting of an execution engine, volatile memory, and nonvolatile storage. The engine is designed for hash algorithms, Rivest–Shamir–Adleman (RSA) key generation, encryption, signing, and random number generation. The chip has a set of special registers called PCRs. These PCRs can be classified into two groups, namely, static and dynamic PCRs, in accordance with their initial value and the time that they can be reset. Static PCRs, PCR 0–16, are reset to 0 on system reboot. Dynamic PCRs, PCR 17–23, are initialized as −1 and 0 at reboot and run-time, respectively. The two PCRs can only be updated through the extend function that aggregates the current content of a PCR with a new content, hashes them, and sends the result back to the PCR. This promising technique can provide two important services, namely, secure storage and platform attestation. In recent years, we have conducted several studies [23–26] on platform attestation and its application. As another technical branch addressing the issue, a trusted environment is isolated in the CPU [27, 28].

The abovementioned studies show that the research on quality control has attracted increasing attention by considering the behavioral traces of workers. However, limited researches have been conducted on verifying the authenticity of behavioral traces. Although hardware-based remote attestation has gained many achievements in other areas, it has limited application in crowdsourcing, especially on the current issue. The results provide references and guidelines for the current research on trustworthy behavioral traces of crowd workers.

3. Problem Formulation

This study considers an issue about ensuring the authenticity of crowd workers' behavioral traces that are used to estimate low-quality answers in crowdsourcing quality control. In the crowdsourcing scenario, employers (called requesters) recruit employees (workers) who complete tasks and earn wages (rewards). The behavioral traces of crowd workers are considered the evidence for detecting low-quality outputs. In this section, this scenario is first modeled to clearly show the current work. Then, we formulate the current problem as a RAP and discuss its security, because the final goal is to design a secure RAP for finding fake behavioral evidence. We present a threat model based on several assumptions.

3.1. Problem Definition. Figure 1 illustrates the basic structure of the proposed model, where the behavioral traces of crowd workers are used to estimate the quality of outputs. In this scenario, requesters submit tasks to the crowdsourcing platform and receive the answer from the platform. The platform assigns tasks to workers using an allocation strategy. Workers perform tasks from the platform and return the outputs back to it. The behavioral traces are captured during completion of tasks and returned to the platform that needs it. The platform estimates the output quality on the basis of the evidence. However, the authenticity of the evidence must be ensured before making a decision. In this study, the authenticity of the evidence refers to its integrity. Three key links, namely, collecting, reporting, and verifying evidence, need to be focused on to obtain authentic evidence. The entire process is formalized as a RAP on the basis of a challenge–response mechanism, as shown in Definition 1.

Definition 1. RAP. The RAP, a triple $(\mathrm{Req}, \mathrm{Res}, \mathrm{Ver})$, consists of three polynomial-time algorithms:

$$c \longleftarrow \mathrm{Req}(n). \tag{1}$$

The challenge-generation algorithm Req takes a random n as input and outputs a challenge c. We write this algorithm as $c \longleftarrow \mathrm{Req}(n)$ because Req may be randomized. In the proposed model, the algorithm is executed by the crowdsourcing platform.

$$r \longleftarrow \mathrm{Res}(s, c) \tag{2}$$

The response-generation algorithm Res takes target's state s and challenge c as inputs and outputs a response r. The algorithm includes the collection of evidence and its report.

$$\tau := \mathrm{Ver}(r). \tag{3}$$

The verification-algorithm Ver takes received response r as input and outputs an authentication token $\tau \in \{0, 1\}$, $\tau = 1$ when the target's state s corresponds to the expected value; else, $\tau = 0$. We write this algorithm as $\tau := \mathrm{Ver}(r)$. because Ver is deterministic. In the current scenario, the process is completed by the crowdsourcing platform. Requesters may also execute the algorithm.

Definition 2. $\mathrm{Exp}_{\Pi}^{\Lambda}$ An adversary Λ submits one state $s' \neq s$ or one challenge $c' \neq c$, accesses $r \longleftarrow \mathrm{Res}(s, c)$, and outputs a response r'. The output of the experiment is defined as 1 when $r' = r$; otherwise it is 0. We write $\mathrm{Exp}_{\Pi}^{\Lambda} = 1$ when the output is 1, and we say that adversary Λ succeeds in this case.

Definition 3. SRAP. We state that a RAP is secure when a negligible function negl exists for any polynomial-time adversary Λ and a sufficiently large n such that $\Pr[\mathrm{Exp}_{\Pi}^{\Lambda}(n) - 1] \leq \mathrm{negl}(n)$.

3.2. Thread Model. To design a secure RAP-based on the trusted anchor, that is, TPM, we assume that adversaries have the following abilities:

(1) Adversaries can eavesdrop on, copy, and replay messages transmitted on channels.

(2) Adversaries can either intercept legal messages or inject forged messages.

(3) Adversaries cannot physically modify the TPM embedded on the motherboard. However, any legal interactions are permitted with the TPM.

(4) Cryptography primitives cannot be broken. In other words, thieves cannot retrieve messages without knowing the key.

As previously mentioned, the function $r \longleftarrow \mathrm{Res}(s, c)$ is valid such that

(1) only the function can compute a valid response r

(2) r must accurately capture the target's state s, that is, $\mathrm{Res}(s', c') = \mathrm{Res}(s, c)$ is negligible for any $s' \neq s$ or $c' \neq c$

Two types of attack may be launched by adversaries on the basis of the above description. One is that the first adversaries simulate $\mathrm{Res}(s, c)$ and correctly compute its output r. Another is that the second is that r cannot correctly reflect (s, c). Adversaries escape the detection of SRAP.

4. SRAP-I

In this section, a specific secure RAP called SRAP-I is designed by the TPM for the scenario of quality control based on workers' behavioral traces in crowdsourcing computing. We first describe the protocol in detail and provide the proof about its security.

4.1. Protocol Description. The protocol includes three phases, namely, integrity measurement, integrity report, and integrity verification.

4.1.1. Integrity Measurement. Interactions occur when performing a crowd task that involves two participants, namely, the agent and the TPM. The agent is responsible for recording the worker's behavior. The TPM is responsible for ensuring the storage security of these behavioral data. The integrity evidence of the worker's behavior is collected and

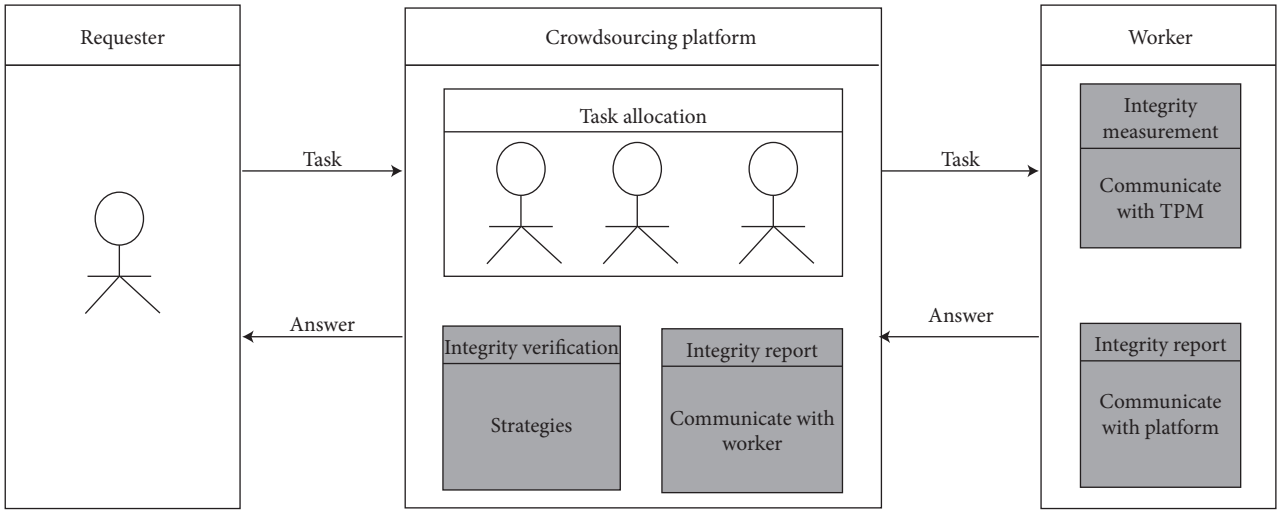

Figure 1: Basic structure of proposed model.

securely stored in the 20th PCR, PCR_{20}, after the phase. Figure 2 shows the interaction steps. In this case, the agent first sends the TPM command, TPM2_PCR_Read, to obtain the initial value of PCR_{20} and write it to the SML. Second, the agent writes an event, e_i, into the storage measurement list, SML, which is a common text file, when the worker makes an action. Then, the agent sends the TPM command, TPM2_PCR_Extend and event, e_i to PCR_{20}. The two operations may be repeatedly executed until the crowd worker completes the task.

4.1.2. Integrity Report. This phase occurs whenever the challenger, the crowdsourcing platform, needs to evaluate the crowd worker. The process involves three participants, namely, the crowdsourcing platform, the agent, and the TPM. Figure 3 shows the interactions among the three participants. The process is a challenge–response protocol. In this case, the crowdsourcing platform first sends a random challenge, nonce, to the agent. Then, the agent sends the TPM command, TPM2_Quote with parameters, nonce and PCR_{20} to the TPM for generating a cryptographic report quote of PCR_{20}. Next, the TPM signs the message that includes the value of PCR_{20} and nonce using the private key of attestation identity keys, AIK_{sk}, and returns the result, quote, to the agent. Finally, the agent generates a response, r that includes the quote, nonce, SML, PCR_{20}, and the public key certificate of attestation identity keys, AIK_{pk}, to the crowdsourcing platform.

4.1.3. Integrity Verification. This process only involves the crowdsourcing platform. The platform first checks the signature of response r using the public key of the TPM, AIK_{pk}, to determine whether it comes from a genuine TPM. Next, it checks nonce and the integrity of the SML by rehashing all its items and comparing the result with the value of PCR_{20}.

4.2. Security Proof

Theorem 1. *SRAP-I is a TPM-based secure RAP with respect to SRAP.*

Proof. As previously discussed, we state that an adversary succeeds or fails when it constructs a response $r' = r$ by taking $s' \neq s$ or $c' \neq c$ as inputs. Here, only integrity report is discussed because two other parts occur in one computer, and we assume that the TPM is reliable.

In the integrity report, c is random, nonce, s is the message {quote, PCR_{20}, SML, AIK_{pk} certificate}, r corresponds to the message, {quote, nonce, SML, PCR_{20}, AIK_{pk} certificate}, the part, quote, is the message $\{\text{nonce}, PCR_{20}\}_{AIK_{sk}}$, and AIK_{sk} is the private part of the attestation identity key of the TPM.

We consider two cases as follows:

(1) $c' \neq c$. In this case, the adversary might generate a new random or use an old random that is used by the challenger, the crowdsourcing platform. The adversary obtains r', {quote$'$, nonce$'$, SML, PCR_{20}, AIK_{pk} certificate}. Apparently, $r' \neq r$ is true. The attack may be launched because nonce$'$ is a plaintext when the adversary attempts to replace nonce$'$ with nonce. Then, the attack is detected because message quote$'$ includes nonce$'$, and quote$'$ cannot be updated without the attestation identity key of the TPM. Similarly, the case using an old random cannot generate r' that is unequal to r.

(2) $s' \neq s$. In this case, the adversary can edit the PCR_{20} and SML because the two parts are unprotected. However, PCR_{20} is embedded in quote and cannot be updated without the attestation identity key of the TPM. Therefore, the adversary cannot construct a fresh $s' \neq s$, making $r' = r$ true.

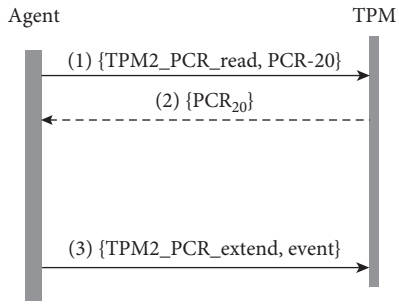

Figure 2: Integrity measurement.

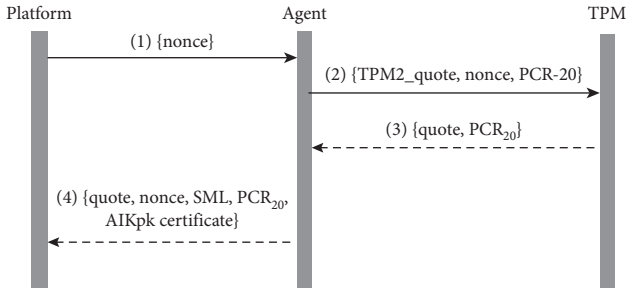

Figure 3: Integrity report.

Thus, SRAP-I is secure with respect to SRAP under a trustworthy TPM.

5. Security Analysis

In this section, we analyze the abilities against attacks, including replay attack, masquerading attack, tampering attack, malicious agent, and software attack to the TPM.

5.1. Replay Attack. In replay attack, the attesting system can replay old messages, and two cases are considered. The first case is that an adversary impersonates the crowdsourcing platform and replays old messages. The second case is that an adversary impersonates the worker and launches replay attacks. Figure 4 shows the interactions between the adversary and the worker. In this case, the adversary replays an old random and obtains an old response, because the protocol does not provide authentication for the crowdsourcing platform. In other words, the adversary succeeds when he or she performs the attack. However, the attack does not cause the genuine crowdsourcing platform to make the correct decision about the integrity of behavioral evidence. Further, the adversary cannot benefit from the attack. Figure 5 shows the interactions between the adversary and the crowdsourcing platform. In this case, the adversary replays an old response, which is an old message, {quote, nonce, SML, PCR_{20}, AIK_{pk} certificate}. The attack can be detected by the crowdsourcing platform, because it can detect the freshness of the response by the random, nonce. Therefore, the adversary fails in the attack.

5.2. Masquerading Attack. In masquerading attack, the adversary masquerades as the crowdsourcing platform or the crowd worker. The adversary can correctly interact with the crowd worker when he or she impersonates the crowdsourcing platform, because the crowd worker does not check the identity of the crowdsourcing platform before building the session, as shown in Figure 6. In this case, the adversary can obtain a fresh response, {quote, nonce, SML, PCR_{20}, AIK_{pk} certificate}, which is meaningless for the adversary. Figure 7 shows the scenario where the adversary plays the crowd worker. This case may occur because the adversary might be a legal crowd worker who also has a valid TPM. However, the TPM is different from the TPM of the crowd worker to whom the crowdsourcing platform wants to talk. Figure 7 shows the interactions between them. The crowdsourcing platform can detect the attack by checking the certificate of the attestation identity key.

5.3. Tampering Attack. In tampering attack, the adversary tampers the response, {quote, nonce, SML, PCR_{20}, AIK_{pk} certificate}. In this case, quote cannot be tampered, because the adversary does not have the corresponding private key. Thus, the adversary only tampers nonce, PCR_{20} or SML. The crowd platform will find the tampering by checking quote when the adversary tampers nonce or PCR_{20}. The crowd platform will find the tampering by rehashing all items in the SML and comparing the result with PCR_{20} when the adversary tampers the SML.

5.4. Malicious Agent. This attack occurs when the agent does not run with the expected behavior. In other words, the agent has been tampered and cannot strictly follow the protocol steps. In this case, only the adversary attempts to rewrite the incorrect behavioral traces, because the correct result is meaningful for him or her. This condition is because the adversary wants to obtain the corresponding pay. However, three difficulties are found. First, predicting the correct behavior traces in solving a task correctly is difficult. Second, the attack can be prevented by checking the platform configurations stored in the PCR. The advantage of using this method is that other malicious software can be found. The disadvantage of this method is that it results in high traffic load. Third, modifying the software is difficult without the corresponding source code. The adversary may prefer to solve the task rather than decompiling the software.

5.5. Software Attack to TPM. This attack involves resetting PCR_{20} without rebooting the system and pushing known good values into PCR_{20}. The adversary must redo all operations as the known sequence of operations to mount the attack successfully. Therefore, mounting the attack for the adversary is meaningless. By contrast, the crowdsourcing platform aims to perform the task in accordance with the current sequence of operations. The adversary will destroy the current protocol when the private key of attestation identity key is retrieved as a software attack to the TPM. It will use the key to sign {nonce, PCR_{20}} and can generate the correct response. However, this process is extremely difficult because the private key is stored in the TPM.

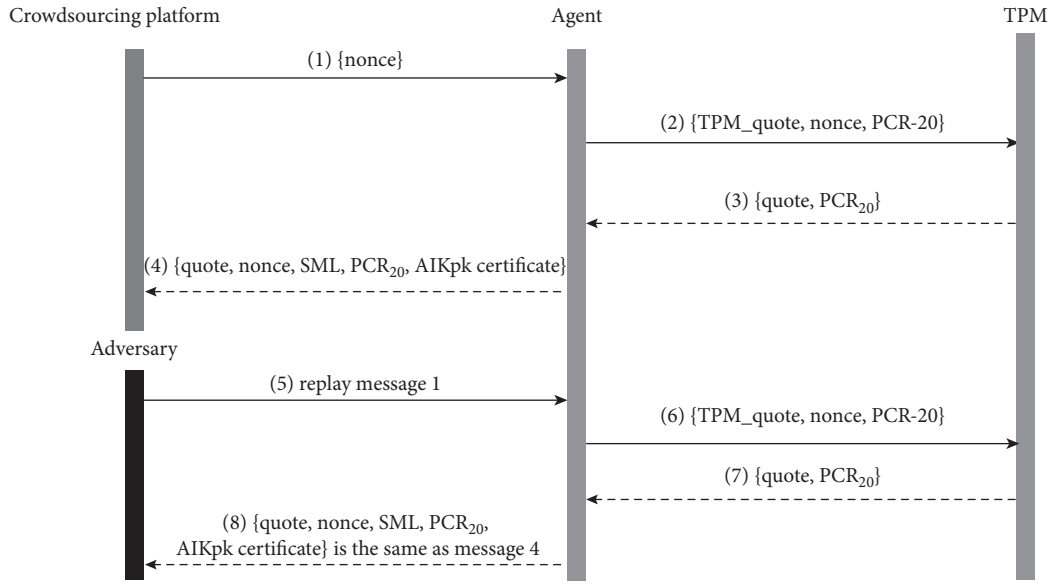

Figure 4: Replay attack on the worker.

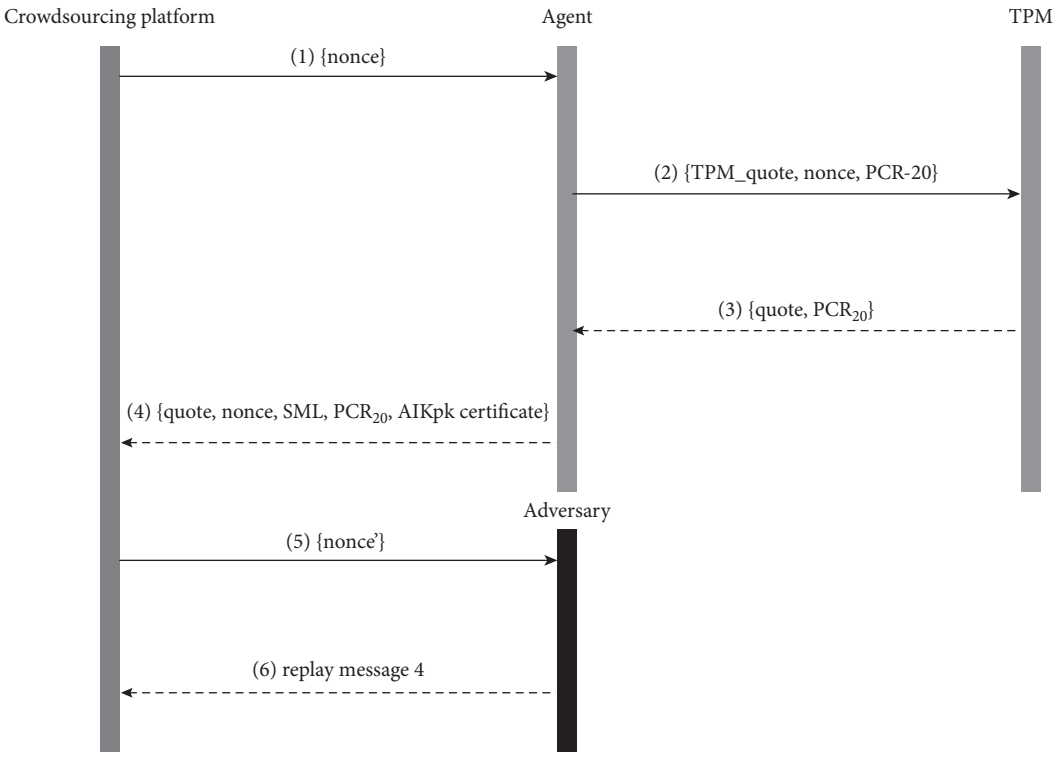

Figure 5: Replay attack on the crowdsourcing platform.

Thus, the current protocol can resist the replay attack, masquerading attack, tampering attack, malicious agent, and software attack to the TPM.

6. Performance Evaluation

In this section, we first introduce two datasets. The first dataset is an open and real dataset of crowd workers' behavioral traces. The second dataset is a customer dataset that is used to record behavioral traces of students when playing an intellectual game. A prototype implementation of SRAP-I is described. We conduct experiments using the selected dataset on the basis of the prototype and results. The evaluation includes two sets of experiments. In the first set, we test the runtime of three main phases, namely, integrity measurement, integrity report, and integrity verification

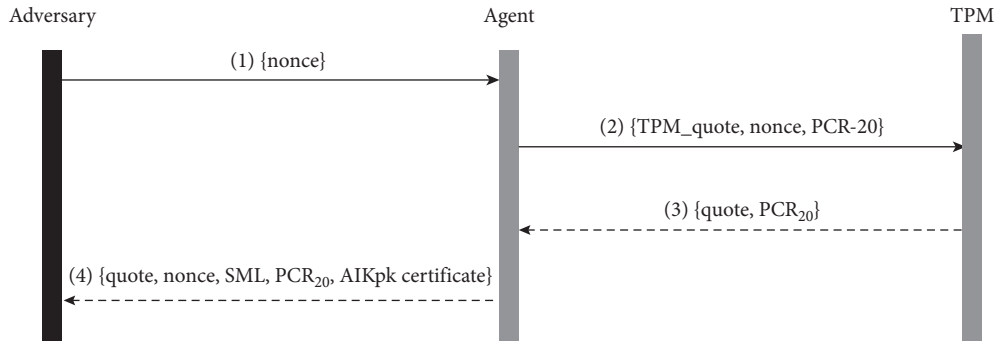

Figure 6: Masquerading the crowdsourcing platform.

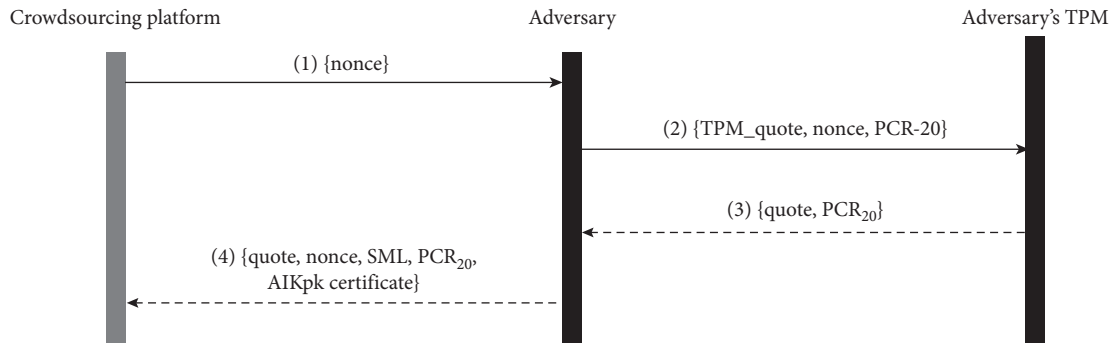

Figure 7: Masquerading the worker performing a task.

using the real dataset. In the second set of experiments, we study their scalability using the custom dataset with different sizes.

The following experiments were conducted based on the computer system. The hardware characteristics of the system include a processor, Intel(R) Core (TM) i7-9700(3.00GHz), a RAM(16.0GB), and a TPM 2.0 device. The software components of the system include Windows 10 operating system and the TPM software stack.

6.1. Dataset of Behavioral Traces. In the current study, two datasets are used to evaluate the current protocol. The first dataset is an open and real dataset of crowd workers' behavioral traces during relevance judging of search results, covering 106 unique workers and 3,984 HITs. The dataset, WebCrowd25K, includes three related parts as follows [29, 30]:

6.1.1. Crowd Relevance Judgments. A total of 25,099 information retrieval relevance judgments are collected on Amazon's MTurk platform. For each of the 50 search topics from the 2014 NIST TREC WebTrack, 100 ClueWeb12 documents are selected to be rejudged (without reference to the original TREC assessor judgment) by five MTurk workers each (50 topics × 100 documents × 5 workers = 25 K crowd judgments). Individual worker IDs from the platform are hashed to new identifiers. Relevance judgments are collected on a four-point-graded scale.

6.1.2. Behavioral Data. For a subset of the judgments, behavioral data characterizing worker behavior are collected in performing relevance judging. Behavioral data are recorded using MmmTurkey [16], which captures various worker interaction behaviors while completing MTurk HITs.

6.1.3. Disagreement Analysis. 1,000 crowd judgments were inspected for 200 documents (five judgments per document, where the aggregated crowd judgment differs from the original TREC assessor judgment), and each disagreement is classified in accordance with the disagreement taxonomy.

In the current study, the behavioral data contained in a JSON file are used. The file contains (key, value) pairs. The key of the JSON object contains mapping IDs, and their values describe the behavior data. This mapping ID must be matched to the "mapping" column of "crowd_judgements.csv" which records the crowd relevance responses for 25k judgments to establish a mapping between HITs and their corresponding worker behavior data.

Another dataset was collected by an intelligent game to test students' creativity. The game asked the subjects to solve the problems that are embedded in game scenarios. The program records their behavior in solving the problems. The dataset includes 3,000 files, and each file corresponds to one student. Most of the files have sizes approximately ranging from 100 kb to 150 kb. We synthesized four files with sizes of 1, 10, 100, and 1,000 kb to satisfy the requirements.

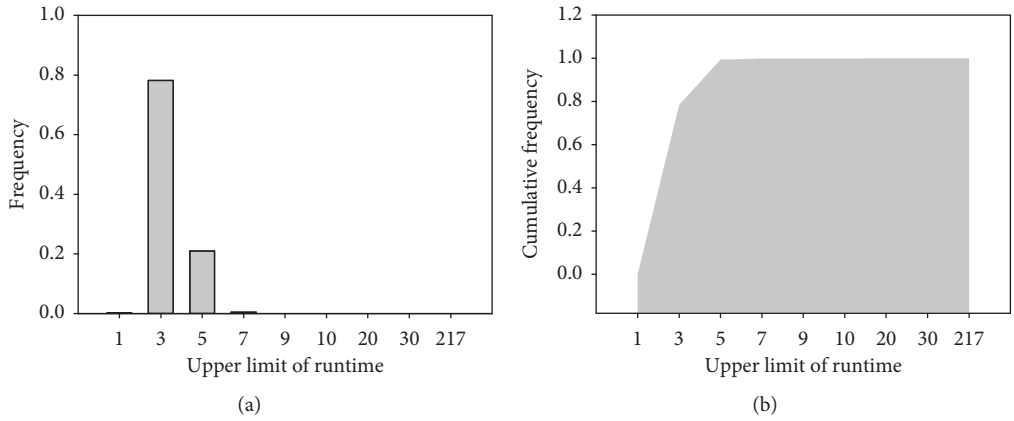

Figure 8: Runtime of integrity measurement. (a) Frequency of integrity measurement. (b) Cumulative frequency of integrity measurement.

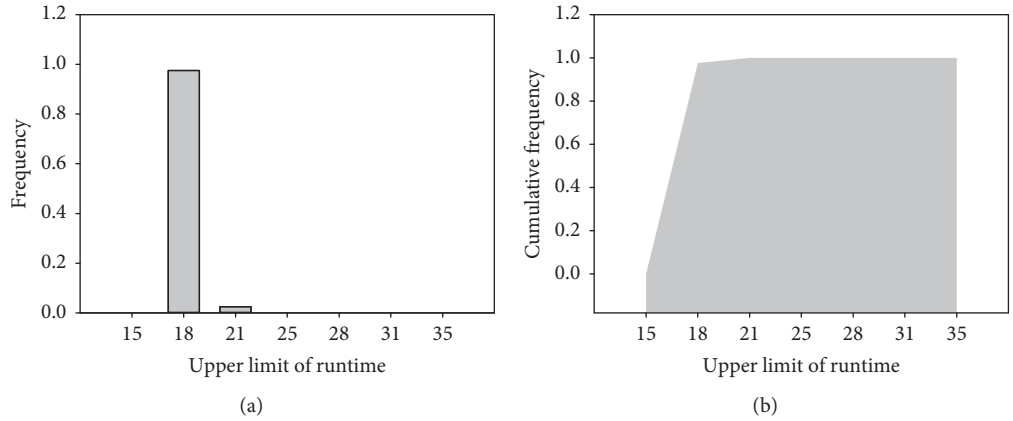

Figure 9: Runtime of integrity report. (a) Frequency of integrity report. (b) Cumulative frequency of integrity measurement.

6.2. Implementation of Prototype. The prototype is implemented on the basis of the TPM software stack implementation for Java language from Microsoft. The library provides abstraction for Windows or simulation for TPM 2.0 devices. In practice, we use a real TPM 2.0 device to provide readers a clear concept of the performance of the current protocol. The prototype can load behavior traces from the JSON file. The main functions are contained in the Java class, TrustAgent, which is implemented by the decorator pattern based on a Java class, Samples, that belongs to the library. TrustAgent provides three main functions, namely, integrity measurement, integrity report, and integrity verification. In the integrity measurement, the JSON behavior traces are first loaded from an external file. Then, the traces are individually hashed and extended to the PCR by the TPM command, TPM2_PCR_Extend, and the SML is generated. In the integrity report, the signed quote is obtained by the TPM command, TPM2_Quote with the parameters, RSA signed key, signed scheme, PCR index, and random challenge, nonce. Then, we obtain the PCR value using the TPM command, TPM2_RCR_Read. Finally, the response, quote, PCR value, SML, and public part of the RSA signed key, are generated and passed to the integrity verification. In the integrity verification, the nonce and the PCR value are first compared with the corresponding parts in the quote. Then, the signature is verified by the public key. Finally, we recompute the hash value of the items in the SML and compare it with the given PCR value.

6.3. Experimental Results. In the first set of experiments, we observed the runtimes of integrity measurement, integrity report, and integrity verification on the basis of the real dataset. A total of 3,984 files of behavioral traces were used to test the runtime. Figure 8 presents the related results about the runtime of integrity measurement.

Figure 8(a) shows that the operation can be completed in 3 ms for most workers' behavior traces. Figure 8(b) shows that the measurement of 99.37% workers' behavioral traces can be completed in 5 ms. The measurement of the first worker's behavioral traces consumes considerable time, which is 217 ms. This finding is because building the session in the TPM is time-consuming.

Figure 9 presents the related results about the runtime of integrity report. Figure 9(a) shows that the operation can be completed in 18 ms for most workers. This finding is the same for 97.54% workers, as shown in Figure 9(b). Figure 10 concludes that 99.87% integrity verification can be

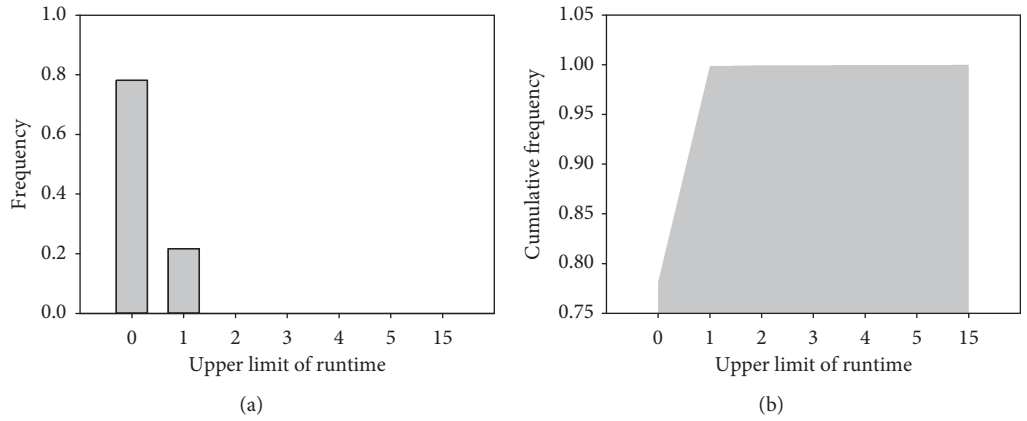

Figure 10: Runtime of integrity verification. (a) Frequency of integrity verification. (b) Cumulative frequency of integrity verification.

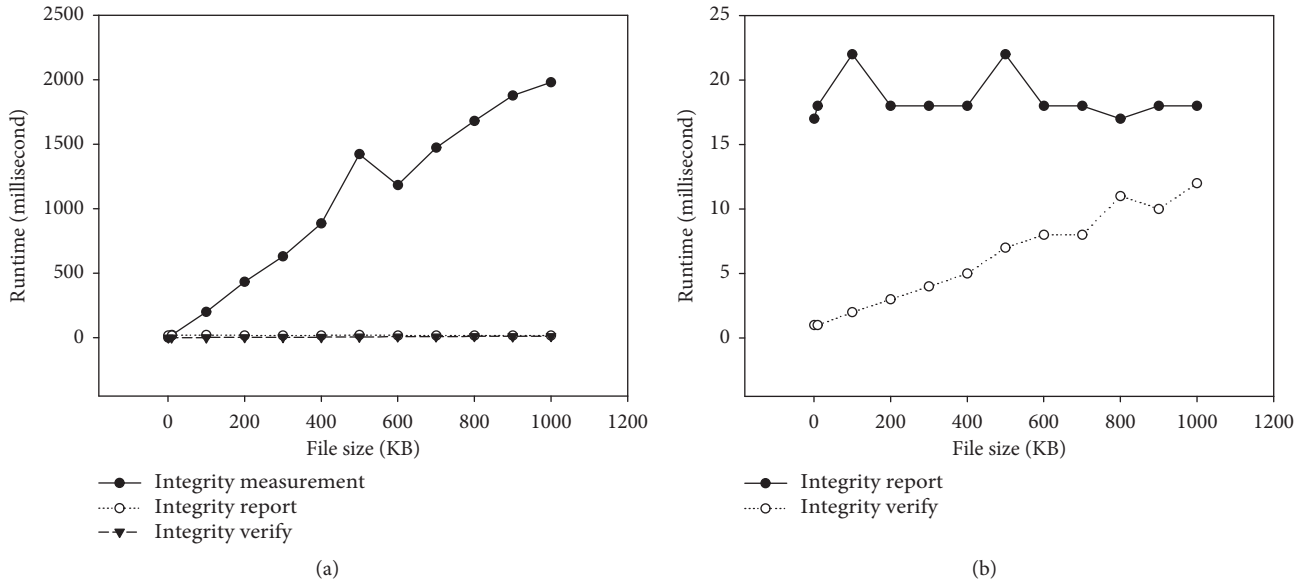

Figure 11: Runtime based on Different Sizes of Files. (a) Three operations. (b) Two operations.

completed in 1 ms. Thus, the operation will not cost the crowdsourcing platform excessive overload. The integrity report is the most time-consuming operation among the three key operations.

In the second set of experiments, we observed the runtimes of integrity measurement, integrity report, and integrity verification as the function of the file scale using the custom dataset. The experiment mainly aims to test the scalability of three main operations, namely, integrity measurement, integrity report, and integrity verification. Figure 11(a) presents the related results about their scalability. Figure 11(b) captures the details of the runtime of the two operations, integrity report and integrity verify. Two observations are obtained. The first observation is that the integrity measurement is the most time-consuming, and the integrity verification takes the least time. The second observation is that the time of integrity measurement continuously increases with the increase in file size, which is opposite the two other operations. Therefore, the scalability of the algorithm is mainly affected by integrity measurement.

7. Conclusion

Crowd behavioral traces have been considered as the evidence that can be used to estimate work quality in crowd computing. Therefore, the integrity of the evidence must be ensured. In this study, we propose a RAP to attest the integrity of the evidence in the crowdsourcing platform. We assume that crowd workers are economical adversaries who make bad behavior to obtain many salaries from the crowdsourcing platform. The TPM, a hardware module embedded on the motherboard, is used as the trust anchor to avoid the economical adversaries. The protocol includes three phases, namely, integrity measurement, integrity report, and integrity verification. The protocol's security is proven and analyzed. The performance is evaluated on the basis of the real and custom datasets.

The experimental evaluation shows that the protocol is an alternative solution for ensuring the integrity of the behavioral evidence. The protocol is secure based on SRAP

and resists attacks, including replay attack, masquerading attack, tampering attack, malicious agent, and software attack to the TPM. The protocol is effective for the real crowd scenario. The conclusion that the integrity measurement will cost considerable times with the increase in file size is acceptable in practice, because the runtime is only 2 s for a 1 mb file. The long-term effects of the protocol in real crowdsourcing platforms will be evaluated in future studies.

Data Availability

The data underlying this article are included within the article.

Disclosure

Any opinions, findings, conclusions, and recommendations expressed in this publication are from the authors and do not necessarily reflect the views of the sponsors.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The authors thank Song Wenai, Liu Zhongbao, Cai Xingwang, and other members of the research group for their suggestions and ideas. This work was supported by the Natural Science Foundation of Shanxi Province of China under Grant 201801D121151.

References

[1] Y. Kryvasheyeu, H. H. Chen, N. Obradovich et al., "Rapid assessment of disaster damage using social media activity," *Science Advances*, vol. 2, no. 3, pp. 1–11, 2016.

[2] D. E. Boubiche, M. Imran, A. Maqsood, and M. Shoaib, "Mobile crowd sensing-taxonomy, applications, challenges, and solutions," *Computers in Human Behavior*, vol. 101, no. 10, pp. 352–370, 2019.

[3] R. Silberzahn and E. L. Uhlmann, "Crowdsourced research: many hands make tight work," *Nature*, vol. 526, no. 7572, pp. 189–191, 2015.

[4] L. V. Ahn, B. Maurer, C. McMillen et al., "reCAPTCHA: human-based character recognition via web security measures," *Science*, vol. 321, no. 5895, pp. 1465–1468, 2008.

[5] T. P. Robinson and M. E. Kelley, "Renewal and resurgence phenomena generalize to Amazon's mechanical turk," *Journal of the Experimental Analysis of Behavior*, vol. 113, no. 1, pp. 206–213, 2020.

[6] F. Daniel, P. Kucherbaev, C. Cappiello, B. Benatallah, and M. Allahbakhsh, "Quality control in crowdsourcing," *ACM Computing Surveys*, vol. 51, no. 1, pp. 1–40, 2018.

[7] Q. Hu, S. Wang, P. Ma, X. Cheng, W. Lv, and R. Bie, "Quality control in crowdsourcing using sequential zero-determinant strategies," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 5, pp. 998–1009, 2020.

[8] C. V. Pelt and A. Sorokin, "Designing a scalable crowdsourcing platform," in *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pp. 765-766, Scottsdale, AZ, USA, May 2012.

[9] P. Michelucci and J. L. Dickinson, "The power of crowds," *Science*, vol. 351, no. 6268, pp. 32-33, 2016.

[10] J. C. R. Licklider, "Man-computer symbiosis," *IRE Transactions on Human Factors in Electronics*, vol. HFE-1, no. 1, pp. 4–11, 1960.

[11] J. Hendler and T. Berners-Lee, "From the semantic web to social machines: a research challenge for AI on the World wide web," *Artificial Intelligence*, vol. 174, no. 2, pp. 156–161, 2010.

[12] J. Howe, M. Tech, and P. Reviews, "The rise of crowd sourcing," *Wired Magazine*, vol. 14, no. 06, pp. 1–6, 2006.

[13] J. M. Rzeszotarski and A. Kittur, "Instrumenting the crowd: using implicit behavioral measures to predict task performance," in *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, pp. 13–22, Santa Barbara, CA, USA, October 2011.

[14] J. Rzeszotarski and A. Kittur, "CrowdScape: interactively visualizing user behavior and output," in *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*, pp. 55–62, New York, NY, USA, October 2012.

[15] P. Heymann and H. G. Molina, "Turkalytics: analytics for human computation," in *Proceedings of the 20th International Conference on World Wide Web*, pp. 477–486, New York, NY, USA, April 2011.

[16] G. Kazai and I. Zitouni, "Quality management in crowdsourcing using gold judges behavior," in *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, pp. 267–276, New York, NY, USA, February 2016.

[17] B. Dang, M. Hutson, M. Lease, and "Mmmturkey, "A crowdsourcing framework for deploying tasks and recording worker behavior on amazon mechanical turk," in *Proceedings of the 6th AAAI Conference on Human Computation and Crowdsourcing (HCOMP)*, pp. 1–3, Austin, TX, USA, November 2016.

[18] T. Goyal, T. McDonnell, M. Kutlu et al., "Your behavior signals your reliability: modeling crowd behavioral traces to ensure quality relevance annotations," in *Proceedings of the Sixth AAAI Conference on Human Computation and Crowdsourcing*, Zurich, Switzerland, July 2018.

[19] U. Gadiraju, G. Demartini, R. Kawase, and S. Dietze, "Crowd anatomy beyond the good and bad: behavioral traces for crowd worker modeling and pre-selection," *Computer Supported Cooperative Work (CSCW)*, vol. 28, no. 5, pp. 815–841, 2019.

[20] R. K. P. Mok, R. K. C. Chang, and W. C. Li, "Detecting low-quality workers in QoE crowdtesting: a worker behavior-based approach," *IEEE Transactions on Multimedia*, vol. 19, no. 3, pp. 530–543, 2016.

[21] A. Martin, "The ten page introduction to trusted computing," 2008.

[22] A. Seshadri, A. Perrig, L. D. Van et al., "SWATT: software-based attestation for embedded devices," in *Proceedings of the IEEE Symposium onSecurity and Privacy*, pp. 272–282, Berkeley, CA, USA, June 2004.

[23] D. L. Fu, X. G. Peng, and Y. L. Yang, "Trusted platform module-based scheme for secure access to outsourced data," *Journal of Electronics and Information Technology*, vol. 15, no. 7, pp. 1766–1773, 2013.

[24] D. L. Fu and X. G. Peng, "TPM-based remote attestation for Wireless sensor networks," *Tsinghua Science and Technology*, vol. 21, no. 3, pp. 312–321, 2016.

[25] D. L. Fu, X. G. Peng, and Y. L. Yang, "Unbalanced tree-formed verification data for trusted platforms," *Security and Communication Networks*, vol. 9, no. 7, pp. 622–633, 2016.

[26] D. L. Fu, X. G. Peng, and Y. L. Yang, "Trusted validation for geolocation of cloud data," *The Computer Journal*, vol. 58, no. 10, pp. 2595–2607, 2015.

[27] G. X. Chen, Y. Q. Zhang, and T. H. Lai, "OPERA: open remote attestation for intel's secure enclaves," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pp. 2317–2331, New York, NY, USA, November 2019.

[28] R. Buhren, C. Werling, and J. P. Seifert, "Insecure until proven updated: analyzing AMD SEV's remote attestation," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1087–1099, New York, NY, USA, November 2019.

[29] T. Goyal, T. McDonnell, M. Kutlu et al., "Your behavior signals your reliability: modeling crowd behavioral traces to ensure quality relevance annotations," in *Proceedings of the 6th AAAI Conference on Human Computation and Crowdsourcing (HCOMP)*, pp. 41–49, Zurich, Switzerland, July 2018.

[30] M. Kutlu, T. McDonnell, Y. Barkallah et al., "Crowd vs. Expert: what can relevance judgment rationales teach us about assessor disagreement," in *Proceedings of the 41st international ACM SIGIR conference on Research and development in Information Retrieval*, pp. 805–814, New York, NY, USA, July 2018.