Crowd Storage: Storing Information on Existing Memories

Jeffrey P. Bigham[†] **and Walter S. Lasecki**[γ]

[†]Human-Computer Interaction Institute
Carnegie Mellon University
jbigham@cs.cmu.edu

[γ]Computer Science, ROC HCI
University of Rochester
wlasecki@cs.rochester.edu

ABSTRACT

This paper introduces the concept of *crowd storage*, the idea that digital files can be stored and retrieved later from the memories of people in the crowd. Similar to human memory, crowd storage is ephemeral, which means that storage is temporary and the quality of the stored information degrades over time. Crowd storage may be preferred over storing information directly in the cloud, or when it is desirable for information to degrade inline with normal human memories. To explore and validate this idea, we created *WeStore*, a system that stores and then later retrieves digital files in the existing memories of crowd workers. WeStore does not store information directly, but rather encrypts the files using details of the existing memories elicited from individuals within the crowd as cryptographic keys. The fidelity of the retrieved information is tied to how well the crowd remembers the details of the memories they provided. We demonstrate that crowd storage is feasible using an existing crowd marketplace (Amazon Mechanical Turk), explore design considerations important for building systems that use crowd storage, and outline ideas for future research in this area.

Author Keywords
crowdsourcing; storage; memory

ACM Classification Keywords
H.5.m. Information Interfaces & Presentation: Misc.

INTRODUCTION

People are increasingly interested in controlling the lifetime of their digital data. Corporate data retention policies set fixed lifetimes on data to reduce costs and liability. Ephemeral photo-sharing services make shared images unavailable after a short amount of time. SnapChat[1] users now share more than 350 million photos per day. Despite this interest, it is difficult to guarantee that information stored in the cloud is really deleted [3, 11][2]. Tying data retrieval to human memory, which decays over time, could be useful for creating ephemeral cloud-based storage mechanisms.

[1]www.snapchat.com
[2]SnapChats were found to be recoverable http://gu.com/p/3fpk8/tw

CHI 2014, April 26–May 1, 2014, Toronto, Ontario, Canada.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-2473-1/14/04..$15.00.
http://dx.doi.org/10.1145/2556288.2557159

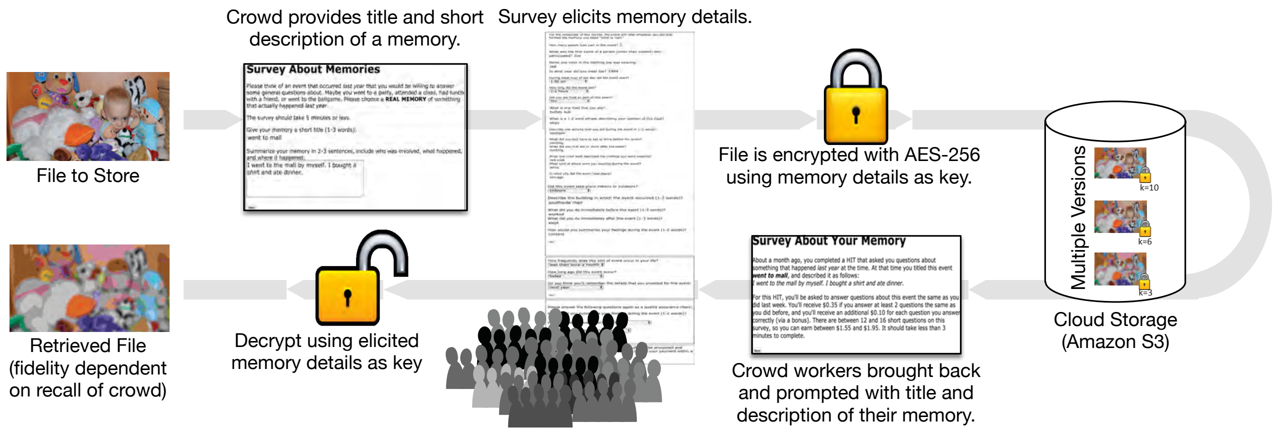
This paper introduces and validates *crowd storage*, the idea that arbitrary digital information (or cryptographic keys allowing access) can be programmatically stored in and recalled from the collective memories of people in the crowd. Digital artifacts stored in this way decay (like human memories do [4]), which allows for parameterizable ephemerality. We began with the idea of storing data directly in the crowd, i.e., crowd members remembered a set of bits, but found that workers were unable to reliably remember data out of context (a similar problem occurs with passwords [12]). Rather than directly storing information, crowd members provide and remember a key derived from their existing memories. Details of your memory of having coffee with a friend may help secure a stranger's photograph.

The WeStore system introduced in this paper uses the same basic cryptographic setup as the Vanish system [3], which allows users to set a lifetime on their data by encrypting the data and then placing the n keys used to decrypt it in the distributed hash table of a large Vuze BitTorrent network. The hash tables periodically remove old data; to decrypt the data at least k of the keys must be retrieved from the network before they disappear. Vanish may be defeated by participating as a node in the network and recording the keys before deletion [11]. WeStore stores its keys not in easily queried public nodes but in the heads of a distributed crowd of people that are much more difficult (and costly) for an adversary to query.

Crowd workers are often treated as anonymous and difficult to bring back, but in practice groups of workers recruited for specific tasks are often quite small. Keeping a consistent workforce allows knowledge to be passed along from one group to the next more easily via *crowd memory* [6]. Prior work has found it useful to retain a particular worker once they have learned how to do a specific task [7]. Crowd storage leverages the memories of crowd workers too, but to store information rather than to support a human computation task.

Our contributions are the following:

- We introduce the concept of *crowd storage* and *WeStore*, a system that demonstrates the feasibility of the idea.
- We illustrate that it is possible to bring crowd workers back over long periods of time (up to 2 years) on a particular existing crowd marketplace (Amazon Mechanical Turk).
- We present the results of a 30-day WeStore experiment in which we store information in the crowd and then extract it again later, including a number of results related to individual components suggesting areas of future research.


Crowd provides title and short description of a memory.
Survey elicits memory details.
File to Store
File is encrypted with AES-256 using memory details as key.
Multiple Versions
k=10
k=6
k=3
Cloud Storage (Amazon S3)
Retrieved File (fidelity dependent on recall of crowd)
Decrypt using elicited memory details as key
Crowd workers brought back and prompted with title and description of their memory.

Survey About Memories

Please think of an event that occurred *last year* that you would be willing to answer some general questions about. Maybe you went to a party, attended a class, had lunch with a friend, or went to the ballgame. Please choose a **REAL MEMORY** of something that actually happened *last year*.

The survey should take 5 minutes or less.

Give your memory a short title (1-3 words):
went to mall

Summarize your memory in 2-3 sentences, include who was involved, what happened, and where it happened:
I went to the mall by myself. I bought a shirt and ate dinner.

Survey About Your Memory

About a month ago, you completed a HIT that asked you questions about something that happened *last year* at the time. At that time you titled this event ***went to mall***, and described it as follows:
I went to the mall by myself. I bought a shirt and ate dinner.

For this HIT, you'll be asked to answer questions about this event the same as you did last week. You'll receive $0.35 if you answer at least 2 questions the same as you did before, and you'll receive an additional $0.10 for each question you answer correctly (via a bonus). There are between 12 and 16 short questions on this survey, so you can earn between $1.55 and $1.95. It should take less than 3 minutes to complete.

Figure 1. The WeStore system for crowd storage. Digital information, e.g. photographs, are input into the system. They are encrypted and then stored in the cloud (S3). The key for decrypting the information is encrypted with all k-combinations of crowd workers' responses to a memory survey. To retrieve the information, a quorum of the crowd workers are brought back to complete the survey again. Information is returned at a fidelity corresponding to how many workers returned and were able to answer the survey the same as they did before.

WESTORE

The WeStore system is a crowd-powered ephemeral storage system in which a digital file is stored, gradually decays, and is eventually lost (Figure 1).

The first component elicits information about existing memories from a crowd worker using a 24-question survey split across 3 pages. The first page asks the worker to think of an experience that happened "yesterday," "last month," or "last year," and give it a short title and description. The timeframe prompt parameterizes the storage, under the hypothesis that the longer someone has already remembered something, the longer they are likely to continue remember it. The second page of the survey asks 20 general questions about the event, such as what city it took place in, the first name of someone else who was there, a description of the weather, what the person did immediately before/after, etc. Importantly, these are recorded as short strings of natural language. They are designed so that many possible answers could be provided, which would make them difficult to guess, although we have not done a formal security analysis. Finally, the third page asks the worker to repeat the answers to two random questions from the prior page and to verify when the event occurred, e.g. "last month." If these are answered incorrectly, the task is rejected and another crowd worker is recruited.

The answers from the survey are then used as keys to encrypt the data that is to be stored. Because we cannot rely on being able to bring every crowd worker back and because we cannot rely on every crowd worker remembering the answers to each question, WeStore uses a storage and retrieval approach that requires only a quorum of crowd workers to remember their part of the key. Just as the rate at which we can bring workers back is likely to be dependent on the source of crowd workers, so is the rate of correct answers likely to be dependent on the question. We can predict these based on prior experience with the questions, although just like human memories, there is no guarantee that past performance will match future results.

The questions that need to be answered correctly also need to be decided in advance of storage. For instance, based on our experience with the system, we found that good results were obtained from requiring at least 5 crowd workers to recall correct answers to 3 specific questions (Figure 2) from the original 20 question list. We leave the problem of setting these more systematically to future work. Each survey answer is then processed to make it easier for the person to reproduce it in the future by removing punctuation, converting to lowercase, removing stop words, stemming each word, and finally removing spaces and vowels, in that order. The key for each worker is the concatenation of the standardized answers for the chosen questions.

Few encryption algorithms available today allow data to be decrypted when only a subset of the keys are provided, and all such algorithms current require all the keys to be carefully generated by a single source [9]. This approach is used by Vanish. For WeStore, we want the keys to be generated by different sources (the crowd workers) and based on their own natural language descriptions of their memories. To support this functionality, WeStore first encrypts the information to be stored using an automatically-generated master key, and then encrypts the master key with every possible correct combination of workers $\binom{n}{k}$, ordered by worker identifier. Although the number of combinations can become extremely large, it stays within reasonable bounds in practice. For instance, if we store a file using 50 crowd workers, expect to bring back 40%, and 30% answer all of the questions correctly, then that means $k \leq 50 \times 30\% \times 20\% = 3$. $\binom{50}{3} = 19,600$. Using AES-256 to encrypt these key combinations took 104 seconds and required 744Kb to store the resulting encrypted keys.

Humans do not generally recall all or nothing of an event, but rather lose detail over time. WeStore can emulate this behavior by storing several versions of the data at different levels of fidelity with increasingly difficult requirements for recalling the higher-fidelity versions. For JPEG images (a lossy format), fidelity is can be controlled by varying the compression rate. We found experimentally that compression rates of between 0.03 and 0.12 yield the largest noticeable effect (levels below or above these look visually similar). Added blur enhances the effect. WeStore can store and encrypt multiple images at different compression levels tied to the numbers of workers required to unlock that level, for instance only 2 workers may be required to unlock the image at compression level 0.03, whereas unlocking the image at compression level 1.0 (no compression) might require 6 workers.

At the time of decryption, WeStore attempts to bring the workers recruited at storage time back, prompts those that reply with their provided memory title and description, and then asks them to complete the survey again. It then constructs each combination of possible keys based on the responses that came in. For Amazon Mechanical Turk, all of the workers recruited at storage time are assigned a qualification so that only they may contribute to unlocking the file at retrieval time. WeStore then sends them an email asking them to complete the follow-up survey, using the provided API.

WeStore assumes that an adversary has access to the cloud storage platform and the source of crowd workers, but not to the list of workers who helped store a specific file or the memory prompts they provided. We further assume that the only way for anyone to access the memory details used as keys is to elicit them from the crowd workers' memories.

EVALUATION

We first conducted a pre-study to validate that Amazon Mechanical Turk workers can be brought back after significant time has passed. To do this we used two data sets of 100 workers that were gathered 2 years prior and 6 months prior to the experiment. We issued 100 HITs to these workers, offering $0.25 to visit the task page and press the submit button (recruited using the process described in the "WeStore" section). Within 24 hours, 21 workers from 2 years ago and 43 workers from 6 months ago had responded.

We then conducted a feasibility evaluation of WeStore with 180 individual crowd workers over the period of 30 days (a period influenced by [4]) in which we stored information in the crowd and recalled it later. In addition to demonstrating the end-to-end feasibility of the system, we also report on our findings regarding individual stages in the WeStore implementation of crowd storage. Workers were offered $0.35 for completing the initial memory elicitation survey and the following retrieval survey. As an added incentive, they received a $0.10 bonus for each question answered the same in the retrieval survey as in the initial survey.

Workers chose a wide variety of different memory types, including the following examples:

- *Jersey Fun - I took a trip to New Jersey with my girlfriend for the weekend. We went to the boardwalk and to the race track where we got to drive exotic cars.*
- *hot and humid - Yesterday I spent a total of 6 hours outside on my property cutting grass with a push mower. My dog, Caesar, could see me from his large, fenced section of yard. It was very humid and I had to change all my clothes once. They were so wet I had to wring them out first.*
- *Attended a Reading - Went to NYC. Saw Neil Gaiman read from his new book. I got a comic signed by him afterwards.*

Thirty days after the storage, we made a retrieval request. Of the 180 workers who originally responded, 81 (45%) responded to the follow-up request within 24 hours. Overall,

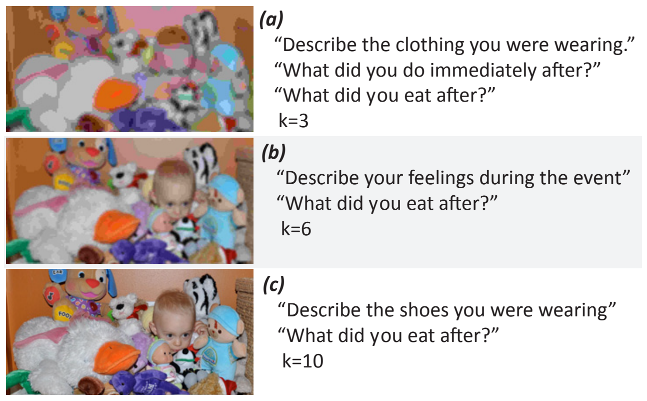

(a)
"Describe the clothing you were wearing."
"What did you do immediately after?"
"What did you eat after?"
k=3
(b)
"Describe your feelings during the event"
"What did you eat after?"
k=6
(c)
"Describe the shoes you were wearing"
"What did you eat after?"
k=10

Figure 2. The effect of different question sets on image fidelity with a given mapping. This particular photo illustrates the impact of fidelity: *(c)* is a perfect reconstruction, *(b)* conveys the "gist" of the image but prevents identification of face, whereas little is conveyed by *(a)*.

30.4% of the answers were correct, as compared to the answers given previously. The accuracy rate for workers in the "yesterday," "last month," and "last year" categories were 28.0%, 28.8% and 33.7%. The pairwise differences between "yesterday" and "last year" and "last month" and "last year" were significant ($F_{1,25}, p < 0.05$). The difference between "yesterday" and "last month" were not detectably significant.

The automated standardization of answers improved overall accuracy by 19.2% (from 25.5% with no standardization to 30.4% with all methods applied). The overall accuracy with methods applied in sequence were as follows: trimming and removing punctuation (25.9%), converting to lowercase (29.5%), removing stop words (30.1%), applying automatic spelling correction (30.2%), stemming each word (30.4%), and finally removing spaces and vowels (30.4%).

Manual coding of the errors revealed that 32.3% of the answers that did not match were semantically similar, but used different words to describe the same concept. For instance, the following answer pairs did not match for the question, "What did you do immediately prior?": *eat* vs. *had dinner*, *relaxed* vs. *relax a while*, and *talked with people* vs. *conversed while waiting*. For the prompt, "Describe the shoes you were wearing," the following pairs did not match: *Black and pink LA Gear shoes* vs. *pink and black tennis shoes*, *Heels* vs. *high heels*, and *clarks desert boots* vs. *Brown Clarks*. Finally, for the question, "What did you eat after?" the following did not match, *milk* vs. *glass of milk*, *soft drink* vs. *soda*, and *water* vs. *Ice Cold Water*. Future work may improve performance by aligning answers like these.

An important consideration for the security of the approach is the distribution of answers provided by the crowd. Overall, 61.1% of the answers that we received were unique. The most repeated answers were "water" in response to what the worker ate or drank before the event, and "blue" and "black" for describing their clothing. Future work may explore how to design questions less likely to result in common answers.

We also explored how the retrieval process could impact fidelity. We restricted ourselves to the first 20 crowd workers to respond (simulating a starting set of 40), then k, the number of workers who correctly match the target answers, varies by the questions that we chose to require. If we map values 3, 5, 10 of k to JPEG compression levels 0.03, 0.10, and 1.0, then the choice of questions parameterizes information retrieval, specifically the retrieval fidelity (Figure 2).

DISCUSSION AND FUTURE WORK

In this paper, we have shown that we can store digital information in the crowd and retrieve it later. We have described design decisions that we made in creating WeStore, which stores information by using details of the existing memories elicited from crowd workers as encryption keys.

Despite what we believe is a commonly held notion, we found that crowd workers (specifically those from Amazon Mechanical Turk) were fairly easy to bring back, which may have implications beyond crowd storage, e.g. in education and training crowd workers to become experts. Future work may explore how different crowds with different properties may be leveraged for crowd storage. For instance, a digital recording of an event may be best remembered by attendees at the event. Digital representations of rich multimedia recordings of such events could be tied to attendees' ability to remember other details of the event [10]. This would build on prior work that has explored how people view the lifetime and legacy of their digital information by artificially adjusting its longevity [4].

While we have shown empirical results of WeStore performance, future work may look at what guarantees can be made. Existing applications of ephemeral storage expect strong guarantees on how long data can be stored and when it will no longer be recoverable, but human memory may be too complex to enforce such strong guarantees. Memories are encoded, stored, and retrieved using a variety of different processes that are understood to varying degrees [1]. Future work may instead make stronger probabilistic expectations of how likely data will be recoverable over time. The retrieval process itself would likely impact this probability as workers who are reminded of events during retrieval may find it easier to remember those events again later. As a result, data stored in this way may only be able to be retrieved once without changing its expected ephemerality. Future work considering these issues may find value in leveraging theories like distributed cognition that describe how memories may already be embedded in the environment beyond the individual [5].

WeStore uses a set of questions that expect natural language answers as keys. An open question is how to design these questions to best support their use as cryptographic keys that both provide reliable access to data and maintain security, especially considering that these two qualities may be at odds. For instance, questions with too many possible answers may make it difficult for crowd workers to provide the same answer again later, whereas a question with too few possible answers may be easy to guess. To avoid brute force attacks, crowd storage relies on there being enough crowd workers and enough memories being elicited to make this approach too costly (or entirely infeasible). It remains to be seen if these qualities will hold in the real world and what support could be provided to help attain them.

Our current conception of crowd storage is dependent on being able to bring back (some fraction of) the same workers that were originally given the information to store. An interesting future direction would be to see how crowd members themselves could pass the storage of information on to others (not unlike the Crowd Physics concept for delivering material goods [8]). For instance, in some formulations a crowd member may be relieved of his or her duty to remember information by finding another crowd member willing to remember that information. This may allow crowd storage to operate robustly at longer timescales (perhaps even longer than the tenure or lifetime of a worker), although may introduce new challenges of coordination and incentivization.

Future work may also consider how the crowd could store not only data but also program instructions that it (or some other group or process) would execute upon some trigger. Just as the Von Neumann architecture of a shared memory space for data and instructions represented an important advance in traditional computing, storing both data and instructions in the collective memory may allow for richer human computation frameworks than available today.

ACKNOWLEDGEMENTS

We thank Jacob Wobbrock and Jeffrey Nichols for their feedback. This work was funded by the National Science Foundation (#IIS-1149709) and by Google.

REFERENCES

1. Baddeley, A., Eysenck, M.W., and Anderson, M.C. Memory. London: Psychology Press. 2009.
2. Caimei, W. and Zhengmao, L., "Self-selecting Sub-secret Keys Sharing Scheme Based on Polynomials over Elliptic Curve," IAS 2009.
3. Geambasu, R., Kohno, T., Levy, A., Levy, H.M. Vanish: Increasing Data Privacy with Self-Destructing Data. USENIX Security Symposium. 2009.
4. Gulotta, R., Odom, W., Forlizzi, J., and Haakon, F. "Digital artifacts as legacy: exploring the lifespan and value of digital data." CHI 2013. 1813-1822.
5. Hollan, J., Hutchins, E., and Kirsh, D. "Distributed cognition: toward a new foundation for human-computer interaction research." TOCHI 2000.
6. Lasecki, W.S., White, S.C., Murray, K.I., Bigham, J.P. "Crowd Memory: Learning in the Collective. CI 2012.
7. Lasecki, W.S., Miller, C., Sadilek, A. Abumoussa, A., Borrello, D., Kushalnagar, R., Bigham, J.P. "Real-time captioning by groups of non-experts." UIST 2012.
8. Sadilek, A., Krumm, J., and Horvitz, E. "Crowdphysics: Planned and Opportunistic Crowdsourcing for Physical Tasks." ICWSM 2013.
9. Shamir, Adi. "How to share a secret," CACM 1979.
10. Toomim, M., Zhang, X, Fogarty, J., Landay, J.A. "Access control by testing for shared knowledge." CHI 2008. 193-196.
11. Wolchok, S., Hofmanny, O.S., Heninger, N., Felten, E.W., Halderman, J.A., Rossbach, C.J., Waters, B., and Witchel, E. "Defeating Vanish with Low-Cost Sybil Attacks Against Large DHTs." NDSS 2010.
12. Yan, J., Blackwell, A., Anderson, R., and Grant, A. "Password memorability and security: empirical results," Security & Privacy, IEEE, 2:5, 25-31, 2004.