For Human Eyes Only: Security and Usability Evaluation

Andreas Pashalidis
KU Leuven and IBBT, ESAT/SCD/COSIC
Leuven, Belgium
{firstname.lastname}@esat.kuleuven.be

Nikos Mavrogiannopoulos

Xavier Ferrer*
Universitat Autònoma de
Barcelona, CVC/IIIA
Bellaterra, Spain
xavier.ferrera@e-campus.uab.cat

Benat Bermejo Olaizola*
Universidad del Pais Vasco
EHU, Department of C.S.
San Sebastián, Spain
bbermejo001@ikasle.ehu.es

ABSTRACT
This paper presents ‘For Human Eyes Only’ (FHEO), our Firefox extension that enables one to conveniently post online messages, such as short emails, comments, and tweets in a form that discourages automatic processing of these messages. Similar to CAPTCHA systems, FHEO distorts the text to various extents. We provide a security analysis of its four default distortion profiles as well as a usability analysis that shows how these profiles affect response time and accurate understanding. Our results illustrate the security/usability tradeoffs that arise in the face of adversaries that use current, off-the-shelf optical character recognition technology in order to launch a variety of attacks. Two profiles, in particular, achieve a level of protection that seems to justify their respective usability degradation in many situations. The ‘strongest’ distortion profile, however, does not seem to provide a large additional security margin against the adversaries we considered.

Categories and Subject Descriptors
D.4.6 [Security and Protection]: Information flow controls; K.4.4 [Computers and Society]: Electronic Commerce—Security

General Terms
Measurement, Performance, Experimentation, Security, Human Factors

Keywords
Privacy, Social networks, Online messaging

*Work done while at KU Leuven, ESAT/SCD/COSIC.

1. INTRODUCTION
The Internet does not forget; emails, blog comments and ‘tweets’, no matter how insignificant or ephemeral, remain online indefinitely. Most of these ‘online messages’ are automatically processed for a variety of purposes. They are, for example, indexed and made available through search engines to an increasingly larger audience. Targeted advertising, censorship, and the identification or profiling of dissidents also benefit from automatic processing of online messages. Some countries, for example, filter specific web pages or even entire websites based on an automatic analysis of their content\(^1\). A recent study on the attitudes of a religious minority in Germany was based on the automatic analysis of over 6700 online forum messages [12]; as is common with psychological experiments that draw data from the Internet [24], it remains questionable whether or not those affected consented to this analysis. Another recent study documents the discomfort caused by automatic message processing; only 9% of respondents found targeted advertising based on email contents to be “ok as long as the email service is free” [19].

This paper presents ‘For Human Eyes Only’ (FHEO), a Firefox extension that addresses the privacy risks stemming from the indefinite retention and automatic processing of online messages. Unlike other browser extensions with similar goals, FHEO does not rely on encryption. It enables individuals to post online messages in a way that discourages automatic analysis, and, in tandem with a message distribution server, ensures that messages disappear from the server after a given period of time. The extension distorts the user’s text and, while it should not be confused with text-based captcha schemes\(^2\), its security relies on the same principles.

Besides presenting it, we also develop a methodology for the analysis of FHEO’s security with respect to three types of adversaries, and proceed with such an analysis. Moreover, we conduct a usability analysis that measures the impact of different distortion levels on the human ability to deal with short messages, and we report the results.

\(^1\)http://opennet.net/research/profiles
\(^2\)CAPTCHA stands for ‘completely automated public Turing test to tell computers and humans apart’ but we write the acronym using lower case characters for readability.
2. RELATED WORK

The service at hidetext.net provides a web interface that generates images from text provided by the user. It does not, however, distort the text, and it remains unclear for how long the images remain online. Moreover, since it is not integrated in the browser, using it for online messaging is impractical. Scramble [3] is a browser extension with a workflow that is similar to the one of FHEO. Scramble encrypts the user’s message and, hence, the user must select a set of recipients for each message and obtain a copy of their public keys. Scramble offers strong security because, assuming that decryption keys are not leaked, message content remains completely inaccessible to both computers and humans. Of course, messages can be automatically processed after decryption, for example by a legitimate recipient’s computer. While FHEO offers only weaker security (namely a certain degree of protection against automatic, and no protection against manual, recovery), it offers increased usability by not relying on keys, user names, or passwords. Moreover, message content remains, at least to some degree, inaccessible to computers, irrespective of whether they belong to legitimate recipients or outsiders. Other systems that aim to protect user-generated content, such as StegoWeb [4] and NOYB [10], are more similar to Scramble rather than FHEO; some interesting and practical ones are discussed in [4].

Similar to text-based captchas, FHEO’s security relies on the fact that humans are currently better at deciphering some types of distorted text than computers. While many captcha schemes are broken (see, for example, [2, 9, 17, 30]), others may still resist recovery attacks [2, 6]. Section 4 explains how FHEO takes into account the lessons learned from these previous works.

In this paper, we use the Tesseract character recognition package [25] in order to conduct a security evaluation of FHEO. Previous studies on captcha security, such as the one by Kalra’s ‘TesserCap’, as well as Ahmad, Yan, and Tayara’s system [2] have also been based on Tesseract. However, while the main focus of the above and other customised attacks against captcha schemes [6, 9, 17, 30] lies in the preprocessing of images, we did not implement such preprocessing or specialised character segmentation logic, for the following two reasons. Firstly, we aim to evaluate FHEO against off-the-shelf technology that is readily available to anyone. Since FHEO does not add noise and clutter to the image background, most typical preprocessing steps would not be required anyway. Instead of implementing our own segmentation, we could use existing, dedicated captcha-breaking software packages, such as the ones described in [2, 6]. While these packages probably yield higher success rates than any given optical character recognition library, they are likely to do so only when confronted with single or double-word images. A significant effort is presumably required in order to enable them to deal with FHEO-generated images more generally. This is because such images typically contain many words spread over multiple lines, may contain significant amounts of punctuation, and because each word is potentially distorted differently. Furthermore, these packages are currently not available to the general public, and hence do not qualify as ‘off-the-shelf’. Secondly, Tesseract naturally deals with multi-line text, and, in many instances, correctly recognises characters even if they touch each other.

In order to evaluate the usability of FHEO, we measure the response time and accuracy when people respond to simple text-based questions that are presented in a distorted fashion. There exists extensive literature that attempts to model the relationship between response times, accuracy, human ability, the difficulty of the question, and other parameters, in similar, but not identical, settings. Some of these works assume that there exists a direct or indirect tradeoff between accuracy and response time (e.g. [13, 28]), and others assume no such dependency [27]. The problem of describing the distribution of response times has also received attention in the literature, with recent studies by Schnipke, Scrams and van der Linden suggesting that the lognormal distribution provides a good approximation [22, 28]. In this paper, we treat response times independently from accuracy, and we do not attempt to verify or disprove previous studies on response time distribution.

For a recent survey of the relevant literature, the reader is referred to van der Linden and Hambleton’s book [29] and Suh’s thesis [27]. Finally, we would like to remark that, while different from ours in several respects, Kay and Terry’s usability analysis [14] is somewhat similar in spirit to ours. More precisely, the authors measured the time that participants spent while examining online agreements when these were enhanced with different levels of visual enhancement. Similarly, our user study involves examining the time participants spend while reading messages, when these messages are distorted to various degrees.

3. FOR HUMAN EYES ONLY

Whenever filling out an online text field, the user can invoke the main FHEO dialog, shown in Figure 1 either through a context menu or a keyboard shortcut. There he can continue editing his text, and immediately sees a preview, as distorted by the currently active distortion profile. The user can select any of the profiles ‘easy’, ‘fair’, ‘good’, ‘insane’, and ‘custom’, where only the latter reveals all the controls between the preview and the ‘upload’ button. Clicking on the current preview generates a fresh one.

We first explain the control options of the ‘custom’ distortion profile. The sliders labelled ‘Perspective’, ‘Wave’, and ‘Zoom’ take integer values from zero to five, and control the intensity of the corresponding distortion. The words ‘inevitable’, ‘optional’ and ‘suffering’ in Figure 1, for example, were distorted with each of these types (intensity three). The slider labelled ‘Be creative!’, which also takes integer values from zero to five, causes consecutive words to be

3Having said this, we performed Lilliefors tests, adapted for lognormality, for the response times of our questions when not distorted (using the data shown in Figure 11), and, after Holm correction, lognormality was not rejected for any question.
merged, characters to be substituted with other, similarly-looking non-ASCII characters, and vowels to be skipped or doubled when they are not at the beginning or the end of a word. These ‘creative’ effects are applied randomly, and with a frequency that is proportional to the current value of the control (the last row of Figure 2 shows some of these effects). If ‘Extra lines’ is selected, then thick lines (as recommended in [6]) spanning one or more words are generated, and if ‘Mix fonts’ is selected, then each word is rendered with a random one out of five fonts. Note that FHEO applies distortion only to long words. The slider labelled ‘Coverage’, which takes values from zero to ten, controls which words are considered long; if, for example, ‘Coverage’ is set to six, then 60% of all words, and in particular the lengthier ones, are considered long. Each long word is then randomly assigned a single distortion (‘Perspective’, ‘Wave’, ‘Zoom’) from those types with a positive intensity setting. Finally, each word is distorted with the chosen distortion type and corresponding intensity. Words that are not long are rendered without any distortion.

The distortion profiles ‘easy’, ‘fair’, ‘good’, and ‘insane’ are constant mappings to the above controls, shown in Table 1. In this table, ‘P’, ‘W’, and ‘Z’ refer to the three distortion types, ‘CV’, ‘CR’, ‘MF’ and ‘EL’ are short for ‘Coverage’, ‘Creativity’, ‘Mix fonts’ and ‘Extra lines’, respectively, and ‘Y’ and ‘N’ mean ‘enabled’ and ‘disabled’, respectively. Figure 2 shows typical distortion results for each profile.

When the user clicks on ‘Upload and Continue’, the current preview image and expiry setting are uploaded to an image hosting server. The server stores the image and its expiry setting are uploaded to an image hosting server. The server stores the image and its expiry time, and assigns a random identifier to the image it sends back to the FHEO extension. Finally, the dialog closes, and the text field that the user was editing is populated with the link under which the image can be retrieved. Since November 2011, FHEO is distributed over the official distribution channel for Firefox extensions, addons.mozilla.org, and is used by a growing user base.

While its details are outside the scope of this paper, it is worth stressing that we also developed and deployed an image hosting server with a number of countermeasures against abuse, such as denial of service or upload of illegitimate content. These countermeasures have enabled us to run the server without any need for authentication, at least until the time of writing. That is, users are not burdened with registration and passwords.

**Table 1: Preset distortion profiles**

<table>
<thead>
<tr>
<th>Profile</th>
<th>P</th>
<th>W</th>
<th>Z</th>
<th>CV</th>
<th>CR</th>
<th>MF</th>
<th>EL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Easy</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Fair</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>Good</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>2</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>Insane</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>2</td>
<td>Y</td>
<td>Y</td>
</tr>
</tbody>
</table>

**Figure 2:** Typical distortion results under the ‘easy’, ‘fair’, ‘good’, and ‘insane’ distortion profile, respectively (top to bottom).

**4. DISTORTION PROFILE DESIGN**

We now turn our attention to lessons learned from previous studies on captcha security. Chellapilla et al. show that machine learning algorithms can easily recognise individual characters, even if they are distorted in multiple ways [7]. Captcha designers should therefore focus on anti-segmentation techniques, i.e. on distortions that make it hard to identify the location of individual characters [2, 6]. Using machine learning techniques, Bursztein, Martin, and Mitchell developed a generic attack that was shown to be effective against a wide range of deployed captcha systems [6]. However, some anti-segmentation techniques resist their attack. In particular, the authors report that extra lines of random length and whose thickness and slope matches that of character fragments, as well as character collapsing, i.e. ensuring that characters touch, or overlap with, each other, are effective anti-segmentation techniques. It is stressed, however, that character collapsing is effective only if both the character size and the length of the captcha are unpredictable, because otherwise an adversary can make a good educated guess on character locations. The authors also recommend to ‘wave the captcha’ in order to increase protection against the removal of extra lines.

Ahmad, Yan and Tayara developed an attack against a particular variant of the Google captcha and the ReCaptcha systems [2], which were the only two captcha system that were attacked without success in [6]. Their attack segments captcha challenges into individual characters by detecting the probable location of some characters based on certain shape characteristics. More precisely, the attack exploits the fact that detectable shapes such as dots, crosses, and loops point towards the location of characters such as \{i,j\}, \{t,z\}, and \{a,b,d,g,o\}, respectively. The location of characters that do not exhibit an exploitable shape, such as \{m,n,l\}, is not detectable by the attack. The authors also propose a methodology for the evaluation of the robustness of a captcha scheme. According to this methodology, a captcha system’s robustness improves if the challenges it produces do not exhibit invariants such as pixel counts and shape characteristics that are strongly correlated with the
challenge text. Of course, characters should also be collapsed because otherwise segmentation becomes easy.

Previous works disagree on whether or not security improves with a larger character set. While Bursztein et al. [6] argue that “increasing the charset does not offer a significant security gain”, Ahmad et al. [2] explicitly recommend a larger one (“A Captcha that uses a small character set is more vulnerable to automated attacks than a counterpart using a large character set.”). This disagreement is, however, not surprising if one considers the difference between the logic of the two attacks: while the segmentation in [6] does not exploit particular character shapes, the one in [2] does.

Most existing works on captcha security agree that noisy or cluttered image backgrounds, as well as colour variations, while impairing readability, generally do not add much security, if any [6, 17]. This is because it is easy to remove most of such noise and clutter using relatively simple image processing algorithms. Existing work also points to the fact that the usage of dictionary words decreases captcha security.

FHEO takes into account the above insights by (a) randomizing the font size for each word (within certain limits), (b) using a different font for each word when ‘Mix fonts’ is enabled, (c) collapsing the characters with intensity proportional to the corresponding intensity setting, (d) merging consecutive words and altering their spelling with an intensity proportional to the ‘Be creative!’ setting, and (e) expanding the character set, also in a manner proportional to the ‘Be creative!’ setting, and in a way that is likely to confuse attacks such as the one in [2]. This is because the character substitutions invalidate some of the assumptions made by the character locating logic of the attack. In particular, they often substitute characters without a dot, a cross or a loop with characters that do have these characteristics, and vice versa. Table 2 shows the character substitution table used by FHEO’s ‘Be creative!’ setting. If a character is selected for substitution, then a random character from the substitution candidates is chosen. Note that all lower-case ASCII characters have substitutions, but the figure shows only a part of the actual table.

The ‘Zoom’ and ‘Perspective’ distortions cause character size and overlap to become unpredictable, and therefore serve as anti-segmentation techniques. The ‘Wave’ distortion, which is recommended in combination with extra lines, is used with slightly more aggressive collapsing than the other two distortion types. Since the length of messages is unpredictable by nature, FHEO follows all anti-segmentation advice in [6]. FHEO produces single-coloured output on a white, clutter-free background.

FHEO could still benefit from randomizing font type and size on the character, rather than the word, level and by ensuring that extra lines stay within the limits of words, and, more generally, by employing an even larger set of anti-segmentation techniques. Nevertheless, we believe that its security benefits from the fact that different words are distorted differently and that users can tailor distortion intensity and coverage to the perceived sensitivity of each message. This follows the spirit of the advice given in [2], which calls for captcha designers to “increase the number of segmentation-resistant mechanisms and variants, text distortion methods, and fonts that a captcha generator supports”. A more comprehensive application of the robustness methodology proposed in [2] to FHEO is left for future work.

Finally, we stress two important differences between text-based captchas and FHEO. On the one hand, while the former can be made immune to dictionary attacks simply by avoiding dictionary words (doing this is, indeed, recommended by previous work), the latter is inherently subject to dictionary-based attacks. On the other hand, while captcha designers must ensure that all characters remain readable, because humans are expected to recognise every single one of them, FHEO can afford to displace, skip or distort beyond recognition some characters, because humans are presumably better than computers in extracting the original meaning from the context.

5. EVALUATION METHODOLOGY

This section presents our evaluation methodology in terms of both security and usability.

5.1 Security

While a captcha may be considered effectively broken if a computer can solve a mere 1% [6] or even 0.01% [2] of instances, the practical impact of such low success rates in the FHEO setting remains unclear. This is because an adversary may be faced with a needles-in-haystack problem; in order to identify a few ‘interesting’ messages from a large collection, and even after having automatically filtered out, say, half of them, manual examination of the remaining half may still be too much work. Of course, this also depends on the adversary’s budget; a recent study found that, on the black market, one dollar buys roughly one thousand manually entered captcha solutions [20].

In this paper, we consider three adversaries against FHEO, namely a ‘full recoverer’, denoted $A_f$, an ‘index generator’, $A_i$...

\[
\begin{array}{|c|c|}
\hline
\text{Character} & \text{Substitution Candidates} \\
\hline
a & \dd a \dd a \dd a \\
b & \dd B \dd B \dd B \\
c & \dd \dd \dd \dd \dd \\
f & \dd \dd \dd \dd \dd \\
h & \dd \dd \dd \dd \dd \\
j & \dd \dd \dd \dd \dd \\
k & \dd \dd \dd \dd \dd \\
o & \dd \dd \dd \dd \dd \\
p & \dd p \dd p \dd p \\
r & \dd r \dd r \dd r \\
t & \dd t \dd t \dd t \\
w & \dd w \dd w \dd w \\
v & \dd v \dd v \dd v \\
x & \dd x \dd x \dd x \\
z & \dd z \dd z \dd z \\
\hline
\end{array}
\]
denoted \( A_i \), and a ‘list watcher’, denoted \( A_w \). In the following, we use the notations \( M \) and \( I(M) \) to denote a text message and a FHEO-generated image containing it, respectively.

\( A_f \) aims to fully recover the original message. That is, on input of an image \( I(M) \), its goal is to output \( M \). While we evaluate \( A_f \)'s effectiveness based on how closely its output matches the original message \( M \), we are not overly strict with this evaluation. In particular, we ignore any errors made with respect to punctuation. More precisely, we consider a ‘noise reduction’ algorithm, denoted \( NR \), which on input of a message \( M \), removes all characters not classified as letters or digits\(^5\). Based on this, we evaluate \( A_f \)'s effectiveness using the ‘normalised’ distance

\[
NLD(A_f, M) = \frac{LD(NR(M), NR(A_f(I(M))))}{LEN(NR(M))},
\]

where \( LD \) denotes the Levenshtein distance function [16] and \( LEN \) denotes a character count function. The closer this distance is to zero, the better the adversary. Roughly speaking, if, say, \( NLD(A_f, M) = 0.2 \), then the adversary gets 20\% of characters wrong, if \( NLD(A_f, M) = 0.5 \), then only half of the characters are correctly recovered, and if \( NLD(A_f, M) \geq 1 \), then the adversary is unlikely to have recovered any message content. Strictly speaking, \( A_f \) may output a long sequence of characters that contains \( M \), and hence may be considered to have recovered \( M \) even if \( NLD(A_f, M) \geq 1 \). This, however, seems to be of no practical relevance in our context and does not happen in our experiments. Hence, the above normalisation suffices for our purposes; the reader is referred to [18] for a more robust approach to Levenshtein distance normalisation.

\( A_i \)'s goal is to render all messages searchable. More precisely, on input of an image \( I(M) \), \( A_i \) extracts a number of keywords for the purposes of indexing. Its effectiveness is evaluated with respect to a ‘keyword extractor’ \( K \), i.e. an algorithm that, given \( M \), outputs a set of keywords. \( K \) must be specified in advance and is assumed to be publicly known. It could, for example, simply output all words in \( M \) (skipping duplicates). \( A_i \) is then evaluated based on how closely its output matches \( K \)'s, using the well-established metrics of precision and recall, given by

\[
\frac{c}{|A_i(I(M))|} \quad \text{and} \quad \frac{c}{|K(M)|},
\]

respectively, where \( c = |K(M) \cap A_i(I(M))| \) denotes the number of keywords that \( A_i \) recovers correctly.

Finally, \( A_w \) is configured with a watch list \( L \) of ‘interesting’ words, and performs a typical binary classification task. More precisely, on input of an image \( I(M) \), \( A_w \) classifies it as either ‘interesting’ or ‘not interesting’. \( A_w \) can be used, for example, to flag suspicious images from a large collection. \( A_w \) is evaluated using the well established metrics of true positive and negative rates. We use Receiver Operating Characteristic (ROC) curves [8] to visualise these rates. Note that ROC curves show the degree by which a given binary classifier outperforms a ‘dummy’ algorithm that classifies instances randomly.

Note that, given \( A_i \), we can construct a ‘trivial’ \( A_w \) which classifies \( I(M) \) as ‘interesting’ if and only if \( A_i \) outputs a keyword in \( L \). This, however, works only if \( K(M) \) outputs at least one word in \( L \) if and only if \( M \) is interesting. It is questionable whether or not this assumption holds in practice because \( K \) is constructed with a general-purpose index in mind, while \( L \) represents more specific interests, and is likely to contain words that are not covered by \( K \).

5.2 Usability

In order to examine the usability of FHEO, we set up an anonymous online survey. Our survey did not focus on how people rate the usability of FHEO’s interface. It neither aimed to establish how willing people are to actually use the system in their online communications, or the level of discomfort caused by FHEO-generated images. Instead it focused on the readability and the understandability of short, FHEO-distorted text messages, as measured by the time needed to respond to such messages (response time), and the proportion of correct answers that are given when being asked about the content of such messages (accuracy).

More precisely, the aim of the survey was to establish the effect of the different distortion profiles on accuracy and response time. Its results are useful in two respects. Firstly, they enable us to contrast the usability (as measured by these metrics) of each distortion profile with its security properties, therefore leading to more informed decisions as to which profile is optimal in a given context. Secondly, they enable us to better focus our attention on particular distortion types and properties when it comes to improving them both in terms of security and usability. The main point is that it makes little sense to enhance distortions with security in mind but without taking into account usability, or vice versa.

From November 2011 to January 2012, we advertised our survey in different departments of several universities as well as in online forums, over social networks, and via mailing lists. No material incentives for participation were offered. Our invitation emails and the survey landing page clearly indicated that distortions should be eliminated while completing the survey. Moreover, the survey interface included a small timer that counted for how many seconds the current question is displayed. During our formative experiments, we aimed to ensure that the location and size of this timer is such that, while reminding the participant that he is being timed, it is not distracting.

Of course, under real conditions, people are not (aware that they are) timed when reading online messages. Our survey participants, some of whom may have felt urged to read quickly, may therefore have taken less time to read each message. It is unclear whether or not such accelerated reading leads to more mistakes because, presumably, survey participants were also generally more focussed compared to their usual Internet browsing. As mentioned earlier, proving or disproving the existence of a speed/accuracy tradeoff is outside the scope of our study. However, since participants were aware that they were timed, our survey does not emulate real FHEO usage conditions and, although we believe that its results provide an estimate of how distortion profiles impact response time and accuracy under certain conditions, these should not be interpreted as directly applying to real FHEO usage. Section 6.2 describes our survey interface and logic in detail.
6. EXPERIMENTAL SETUP

This section presents our experiments; the next section, in particular, explains our security experiments and Section 6.2 our online usability survey.

6.1 Security

This subsection explains our adversary implementations and the datasets that drove our security experiments.

6.1.1 Full Recoverer

We implemented an adversary $A_f$ based on Tesseract version 3.0.1, a state-of-the-art optical character recognition library with a long history [25]. Tesseract recognises text stretching multiple lines and its adaptive classifier can be trained with custom characters, fonts and a growing number of languages [26]. It should be noted that, while using a dictionary internally, Tesseract consults this in combination with visual hints and that, therefore, its output is not restricted to dictionary words. While it is possible to increase the influence of the dictionary by modifying Tesseract’s source code, we did not do this.

Given an image $I(M)$, our $A_f$ simply invokes Tesseract and outputs whatever string the library recovers, after removing control characters and superfluous spaces. Our adversaries $A_i$ and $A_w$, described below, are using $A_f$ as a subroutine.

6.1.2 Index Generator

We implemented a simple keyword extractor $K$ that, given a message $M$, outputs all words in $M$, except the words that (a) have fewer than three characters, (b) are on a particular list of very common words (provided in the extended version of this paper [21]), and (c) are duplicates. We stress that $K$ does not consult any dictionary.

Given an image $I(M)$, our index generator $A_i$ first invokes $A_f$ and obtains the recovered message $M'$. It then passes each word in $M'$ of at least three characters in length through a dictionary. More precisely, if the word is known to the dictionary, then $A_i$ leaves it intact. Otherwise, it queries the dictionary for a correction. The dictionary returns a correction recommendation along with a confidence $\kappa \in [0,1]$. If $\kappa \geq \tau$, where $\tau \in [0,1]$ is a given ‘confidence threshold’, then $A_i$ replaces the word with the dictionary’s recommendation. Otherwise, $A_i$ deletes the word. In this way, $A_i$ constructs the ‘corrected’ message $M''$, and finally outputs $K(M'')$.

In our experiments, we used two dictionaries, namely the well-known aspell dictionary available at aspell.net, and a custom one which we implemented ourselves. aspell does not support the notion of confidence and hence we assume $\kappa = 1$ in all instances. Our custom dictionary is initialised with one or more word lists, and remembers word frequencies, i.e. how many times each word has been encountered in the lists. When asked to provide a correction for a misspelled word, it first constructs the list of words $w_1, \ldots, w_n$, with minimum Levenshtein distance from the given word. It then recommends the word $w_i$ with the highest frequency and, in case of multiple matches, $w_i$ is chosen based on a character frequency analysis. If even this does not break the remaining ties (in our experiments this never happened), it picks a random $w_i$ from the remaining candidates. In any case, the reported confidence is $n^{-1}$.

Our custom dictionary also tries to split long words, by treating them as two separate words and trying all plausible split points. A misspelled long word is replaced by two words rather than by a single correction only if the product of their confidences exceeds the confidence of the single-word correction (and, of course, $\tau$). Note that aspell also recommends two-word corrections in some cases.

We used two variants of our custom dictionary in our experiments. The first one, which we call the ‘normal’ variant, was initialised with the shortest two lists in Atkinson’s collection of common English word lists, version 7.1. The two lists contain 4442 and 12564 words, respectively. Our second variant, which we call the ‘Twitter’ variant, was initialised with Atkinson’s shortest list and a ‘Twitter’ word list which was derived from the Twitter network, as follows. In January 2012, we subscribed to the Twitter stream and divided each encountered tweet into ‘bad’ and ‘good’ words; a word was considered bad if (a) it is the word ‘RT’ (which means ‘re-tweet’ and occurs frequently), (b) starts with an at sign (username, which also appear frequently, start with this character), (c) starts with ‘http’, or (d) contains non-printable characters. All other words were considered good. Tweets with no good words, and tweets with more than twice as many bad than good words were discarded. Given that many different languages appear on the Twitter stream with a significant minority containing a mixture of multiple languages, we tested whether or not the remaining tweets contained sufficiently many English words. In particular, we discarded tweets with less than 25% of their good words appearing on Atkinson’s shortest word list. The good words of the remaining tweets, without their surrounding punctuation, were added to the Twitter list which, in the end, contained 102990 words (many of which are duplicates).

Note that the Twitter stream is a random sample of all current tweets on the Twitter network, and that our language test filtered out about 76% of the stream.

6.1.3 List Watcher

Given a watch list of words $L$, an image $I(M)$, and a discrimination threshold $t \in [0,1]$, and after obtaining the recovered message $M'$ from $A_f$, our list watcher $A_w$ classifies $I(M)$ as ‘interesting’ if and only if $LD(w, l) \leq t \cdot \text{LEN}(l)$ for any pair of words $(w, l) \in M' \times L$, where $LD$ denotes Levenshtein distance, and $\text{LEN}$ denotes a character count function. Note that $A_w$ does not use any dictionary.

6.1.4 Datasets

Our experiments were driven by two test datasets, where each dataset is simply a collection of text messages. Our first dataset, called the ‘Peter Pan’ (PP) dataset, was derived from Barrie’s homonymous novel. In particular, we downloaded the text and, after filtering out chapter names, front/back matter, as well as all double quote characters, we divided it into sentences.

Our second dataset, called the ‘Twitter’ (TW) dataset, consists of a collection of tweets which we obtained from the Twitter stream. We ‘sanitized’ tweets by dividing them into ‘good’ and ‘bad’ words, and by discarding tweets with

---

1[http://code.google.com/p/tesseract-ocr]
2[http://wordlist.sourceforge.net]
3[http://dev.twitter.com/docs/streaming-api]
4[http://www.gutenberg.org/cache/epub/16/pg16.txt]
Each question was further distorted using a profile, chosen randomly per participant and question, from the profiles ‘none’, ‘easy’, ‘fair’, ‘good’, and ‘insane’. The profile ‘none’, which we used as a baseline in our evaluations, does not perform any distortion. A script running in the browser measured the response time, i.e. the time between the question fully appearing on the screen, and the participant clicking on one of the given links. Before proceeding with the next question, our survey server recorded the ‘response’, which includes (a) the static identifier of the question that was just answered (not shown to the user), (b) the profile under which the question was distorted, (c) whether the answer was correct, wrong, or ‘skipped’, and (d) the response time reported by the client-side script. If a question was answered multiple times (this could happen, for example, if the participant used the browser’s ‘back’ button), our server recorded only the first response. It also kept track of which responses correspond to each participant, and, using a cookie, prevented participants to complete the survey multiple times. At the end of the survey, participants were given the opportunity to provide feedback. Based on this feedback, we decided to discard the first response from each participant, because this incorporated delays due to the participant getting acquainted with the survey logic and its interface.

Between November 2011 and January 2012, we gathered 7824 responses from 271 distinct participants. However, since online surveys are not taken seriously by everyone, we discarded records that do not seem to represent a genuine effort to complete the survey, as well as responses that were not usable for some other reason. In particular, we discarded responses indicating that the question was skipped. From the 200 responses that were skipped, 30 out of 1636 (1.83%), 27 out of 1659 (1.63%), 29 out of 1315 (2.12%), 36 out of 1242 (2.90%), and 78 out of 1701 (4.59%) correspond to questions that were distorted under the ‘none’, ‘easy’, ‘fair’, ‘good’, and ‘insane’ profiles, respectively. Then we discarded all responses of participants who had (a) answered fewer than six questions that were either ‘none’ or ‘easy’-distorted, and (b) answered more than 20% of ‘none’ and ‘easy’-distorted questions incorrectly. This ensures that participants who merely had a casual look at the survey, i.e. did not respond to sufficiently many questions (73 participants), or randomly selected answers, are not represented in our results. Note that a random answer is correct approximately 33% of the time. Our threshold of 20% is stricter because our questions are very easy; indeed, this filter removed only eight participants, of which four had more than 82% of their answers incorrect (under ‘none’ or ‘easy’ distortion), suggesting that incorrect answers were chosen purposefully. One participant, whose responses indicated zero response times, was also removed; this participant probably had JavaScript disabled.

Although our survey landing page indicated that distractions should be eliminated while completing the survey, in order to nevertheless identify distraction-induced outliers, we plotted the response time of all responses that correspond to each distortion profile. Using these plots, we identified and discarded five responses that had a disproportionately large response times (above 130 seconds). We are very conservative with the removal of outliers because we expect that most users who either did not understand a question, or knew that they were responding with a large, distraction-induced delay, simply selected to ‘skip’ the question.

### Table 3: Average and Standard Deviation of character, word and keyword counts of our datasets

<table>
<thead>
<tr>
<th></th>
<th>Char. count</th>
<th>Word count</th>
<th>Keyword count</th>
</tr>
</thead>
<tbody>
<tr>
<td>PP Avg.</td>
<td>73.7</td>
<td>17.8</td>
<td>8.3</td>
</tr>
<tr>
<td>Std. Dev.</td>
<td>55.3</td>
<td>13.2</td>
<td>6.0</td>
</tr>
<tr>
<td>TW Avg.</td>
<td>47.0</td>
<td>11.4</td>
<td>6.0</td>
</tr>
<tr>
<td>Std. Dev.</td>
<td>26.0</td>
<td>6.4</td>
<td>3.2</td>
</tr>
</tbody>
</table>

**Figure 3: Survey interface**

![Survey interface](image)

insufficiently many English words, just as we did for the construction of the Twitter word list for our index generator (see Section 6.1.2). The good words of each remaining tweet, in their original capitalization and with their surrounding punctuation intact, served as our Twitter dataset, each sanitized tweet as a separate message. We stress that, in adherence to best practices from the machine learning community [5], this test dataset was derived from different tweets than the adversary’s dictionary in Section 6.1.2.

The PP and TW datasets contain a total of 2646 and 10000 messages, which are composed of 47231 and 114157 words, or 250909 and 613589 characters, respectively. Their average per-message character, word and keyword (as determined by K) counts are shown in Table 3.

### 6.2 Usability

In order to examine the human ability to understand short, FHEO-generated messages, we set up an anonymous online survey that proceeded as follows. After providing demographic data such as their gender, age, as well as general education and English level, participants were asked to respond to 40 fixed messages, each containing a multiple-choice question using the interface shown in Figure 3. We crafted these messages to be of moderate length, roughly imitating the length of short but still informative comments or status updates typically found on social networks and blogs. Each question was about content which is included in the same message. We also carefully selected the answer options in a way that requires one to truly understand, or at least fully read, all parts of the message. That is, all answer options were identical or at least closely resembled words that appear in the message. In the following, we refer to our message/questions simply as ‘questions’ (see the extended version of this paper for the full list of questions and answer options [21]).

Both the questions and their corresponding answer options were presented to each participant in a random order.
Overall, 6691 responses from 182 participants survived our filters and are used as a basis for our results. Figure 4 shows the age and gender distributions of these participants. 2 (1%), 16 (8%), 59 (31%), and 112 (59%) participants declared an education level of ‘basic’, ‘secondary’, ‘bachelor’, and ‘postgraduate’, respectively (see Figure 5). Similarly, 0 (0%), 33 (17.5%), 125 (66%), and 31 (16.5%) declared an English level of ‘beginner’, ‘intermediate’, ‘advanced’, and ‘native’, respectively.

Approximately 100 participants left some feedback after completing the survey. The number of participants that claimed all distortions to be readable is approximately equal to the participants who ‘complained’ about “some messages” being too hard to read. Several participants left a comment along the lines of “everything is still quite readable, except maybe for the strongest settings”. While some participants found the survey to “be fun”, a larger number found it to be rather tedious because they felt it was overly lengthy. A couple of participants reported that they were ‘surprised’ how, after a few questions, they could figure out the writing fairly quickly. Finally, two comments drew our attention to the fact that people with disabilities, in particular the blind and those suffering from dyslexia, are likely to face difficulties when confronted with FHEO-generated images.

7. EVALUATION RESULTS

This section presents our security and usability evaluation results. Note that these results refer to the current FHEO version (1.04), but also apply to all versions from 1.02 upwards; future versions, however, may use different default distortion profiles. In this case, a new analysis will be required.

7.1 Security

Our security experiment proceeds by distorting each message $M$ in a given dataset using a particular profile, and subsequently feeding the generated image $I(M)$ into each of our three adversaries described in Section 6. We conducted a separate experiment set for each of our two datasets.

Figure 6 and Table 4 show the NLD that $A_f$ yields for the different distortion levels. We observe that, even under ‘none’-distortion, the NLD is above zero, indicating that there is room to improve Tesseract. We also see that, for all distortion profiles, the obtained NLD values are essentially identical for both datasets. This indicates that Tesseract deals with both types of text equally well. Finally, note that, even under ‘insane’ distortion, NLD is only around 0.75 on average, suggesting that a large amount of characters (approx. 25%) is still recovered correctly. This seems to be because FHEO does not distort every word, but only a proportion in accordance to the coverage parameter.

Note that Tesseract’s default configuration is based on the English language and can recognise a variety of Latin character glyphs. We spent some effort training Tesseract with slightly distorted characters taken FHEO-generated images. This did not improve $A_f$’s NLD values; since our aim is to investigate FHEO’s security against off-the-shelf technology, we did not invest further in training Tesseract.

Experiments with $A_i$ were conducted both with aspell and our custom dictionary, as described in Section 6.1.2. In the PP and TW dataset, we used the ‘normal’ and the ‘Twitter’ variant of our custom dictionary, respectively. Moreover, in both experiments we also evaluated $K(A_f)$, i.e. the same adversary but without any dictionary processing. Figures 7 and 8 show the resulting precision/recall graphs. Our custom dictionary was tested with all confidence thresholds $\tau \in \{0.00, 0.05, \ldots, 1.00\}$, and hence it is represented by multiple points per distortion profile. Interestingly, our re-
results show that, under ‘none’-distortion, dictionary processing actually reduces recall in both datasets, and precision improves only marginally with aspell in the PP dataset. This is because some keywords are not included in the dictionary, yet the processing forces \( A_i \)’s output to contain exclusively dictionary words.

Figures 7 and 8 also show that, while dictionary-based processing only marginally improves precision and recall under ‘easy’ and ‘insane’ distortion, under ‘fair’ and ‘good’ distortion, these improvements are substantial. The figures also show the limits within which \( A_i \) can be tuned to a particular precision/recall tradeoff by means of the confidence threshold parameter of our custom dictionary. Finally, observe that, while aspell outperforms our custom dictionary (normal variant) under some distortion profiles in the PP dataset, our custom dictionary (Twitter variant) consistently outperforms aspell in the TW dataset. This highlights the importance of domain knowledge when post-processing recovered text with a dictionary.

Figures 9 and 10 show the ROC curves for \( A_w \) when initialized with \( L_{PP} = \{ \text{wendy, peter, michael} \} \) and \( L_{TW} = \{ \text{girls, money, morning} \} \) and evaluated against the PP and TW datasets, respectively, and Table 5 shows the corresponding equal error rates. Note that 695 of the 2646 messages in the PP dataset, and 295 of the 10000 messages in the TW dataset actually contain a word in \( L_{PP} \) and \( L_{TW} \), respectively. Also note that, while \( A_w \) directly achieves only the data points shown in the figures, all points on the curves are realisable [23].

Observe that moving from ‘easy’ to ‘fair’ distortion, as well as from ‘fair’ to ‘good’ distortion, substantially increases protection against automatic recovery. However, the extra security margin offered by ‘insane’ distortion is rather limited. This is because, in both datasets, ‘good’-distorted messages are misclassified already very often, not leaving much room for even higher error rates. Also note that, with the exception of ‘none’ distortion, error rates are consistently higher in the TW than they are in the PP dataset. This suggests that the number of messages that contained words similar to the words in the watch list was higher in the TW case than in the PP case. That is, the performance of \( A_w \) depends on both the watch list and the dataset.

As an example, we contrast \( A_w \)’s performance under the ‘none’ and ‘fair’ distortion profiles. Under ‘none’ distortion, \( A_w \)’s equal error rate (EER) of 2.0% in the PP dataset means that it correctly identifies 681 of the 695 messages as ‘interesting’, while also, incorrectly, identifying 39 of the remaining 1951 messages as such. In the TW case, where the EER is 1.7%, \( A_w \) correctly identifies 290 of the 295 interesting messages as such, while also misclassifying 165 from the remaining 9705 messages as ‘interesting’. Under ‘fair’ distortion, the performance of \( A_w \)’s performance is much poorer: the EER of 28.7% in the PP dataset means 496 of the 695 messages are ‘correctly identified, but also, incorrectly, identifying 165 of the remaining 9705 messages as ‘interesting’. The EER of 38.2% means that only 182 out of the 295 messages are correctly identified as ‘interesting’, while 370 of the remaining 9705 messages are also flagged as ‘interesting’.

### 7.2 Usability

Although our survey questions may appear to be of similar difficulty, Figure 11 reveals that, under ‘none’ distortion, some questions were answered much faster than others. While 19 out of the 40 of questions, for example, yielded a median response time between 7.9 and 9.6 seconds, the median response time for question 24 (14.2 sec) is more than twice that of question 0 (6 sec). Since these values do not seem to be normally distributed (Lilliefors normality tests [15] yield Holm-adjusted \( p < 0.05 \) for 17 questions), we used a Kruskal-Wallis test in order to determine significance of the above differences. The test finds differences to be significant (\( p \ll 0.01 \)). Note that our finding that re-
response times are not normally distributed agrees with previous studies (see Section 2).

A similar analysis reveals that, while 28 out of the 40 questions are answered correctly at least 98% of the time under ‘none’ distortion, the accuracy for 6 questions was less than 95%. Binomial tests, which we performed on a few selected question pairs, suggest that these differences are significant, too (Holm-adjusted \( p < 0.05 \)). Hence we compute the per-distortion profile average response time and accuracy for each question separately. These values are visualised in Figure 12 and their average and standard deviation are shown in Table 6. (Note that the full list of values are provided in the extended version of this paper [21].) We observe that, as expected, the average response time increases and the average accuracy decreases as distortions become more heavy. Moreover, as distortion intensity increases, so does the variance of these two quantities. This suggests that the ability to deal with distorted text varies among people to an extent that is proportional to the distortion intensity.

Since our survey participants needed, on average, approximately 11.2, 14.9 and 16.6 seconds to respond to a question when distorted under the ‘fair’, ‘good’ and ‘insane’ profiles, respectively, these distortion profiles induced an increase in response time of approximately 10%, 46%, and 63% when compared to the response time of ‘none’-distorted messages (10.2 seconds). Under ‘easy’ distortion, participants were on average almost 0.2 seconds faster than under ‘none’ distortion, without any loss in accuracy. The average accuracy of participants remained essentially the same (approximately 98%) under ‘none’, ‘easy’, and ‘fair’ distortion, and then dropped by about 5.3% under ‘good’ distortion. There was a further decrease of approximately 1% under ‘insane’ distortion.

We wish to establish the statistical significance of the above effects that distortion levels had on response times and accuracy, while eliminating, as much as possible, the influence of the inherent difficulty of each question. Therefore, and since our survey follows a randomized block design [11], we performed two Friedman tests [15], one for response time and one for accuracy; both tests indicate significant differences (\( p \ll 0.01 \)) between the profiles.

A post hoc analysis with pairwise Wilcoxon signed-rank tests between distortion profile pairs shows that, in terms of response times, all distortion profile pairs are significantly different (Holm-adjusted \( p \ll 0.01 \)), except the pair ‘none’-‘easy’ (Holm-adjusted \( p \approx 0.61 \)). This suggests that our finding that participants respond to ‘easy’-distorted questions faster than they do to ‘none’-distorted ones, is not significant.

Similar post hoc tests on the accuracy results suggest significant differences (Holm-adjusted \( p \ll 0.01 \)) between all distortion profile pairs except the pairs ‘none’-‘easy’, ‘none’-‘fair’, ‘easy’-‘fair’, and ‘good’-‘insane’ (Holm-adjusted \( p \approx 0.90, p \approx 0.98, p = 1, \) and \( p \approx 0.79, \) respectively). The main finding here is the difference between the pair ‘fair’-‘good’, where the average accuracy of survey participants is 97.9% and 92.6%, respectively.

8. DISCUSSION

Based on the results presented in the previous section, we draw the following conclusions. The distortion profile ‘easy’ appears to be as usable as plain text, i.e. as if no distortion is applied at all. However, as expected, it offers little protection against automatic recovery.

The ‘fair’ profile provides some protection against automatic recovery. Depending on the scenario and adversary type, this degree of protection may or may not be sufficient. In terms of usability, this distortion profile increases response times only slightly without impacting accuracy. In situations where even a moderate usability degradation is costly and protection against automatic recovery is not critical, this profile is likely to be optimal.

The ‘good’ profile consistently provides more protection against automatic recovery than the ‘fair’ profile, against all adversary types we considered. We expect that, in many scenarios, the degree of protection offered by the profile, is sufficient. In terms of usability, this distortion profile moderately increases response times, and also induces a slight drop in accuracy. However, since our survey questions were answered with approximately 93% accuracy under this distortion profile, we believe that, currently, the security benefits of this profile outweigh its usability disadvantages.

The distortion profile ‘insane’ induces a rather large increase in response times (in our survey more than 60%), while not offering a much higher degree of protection against

---

**Table 6: Average and Standard Deviation of response time (RT) in seconds and accuracy (ACC) under different profiles**

<table>
<thead>
<tr>
<th>Distortion Profile</th>
<th>RT Average (seconds)</th>
<th>Std. Deviation</th>
<th>ACC Average</th>
<th>Std. Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>10.17</td>
<td>1.97</td>
<td>0.980</td>
<td>0.029</td>
</tr>
<tr>
<td>Easy</td>
<td>9.99</td>
<td>1.84</td>
<td>0.978</td>
<td>0.032</td>
</tr>
<tr>
<td>Fair</td>
<td>11.17</td>
<td>2.56</td>
<td>0.979</td>
<td>0.050</td>
</tr>
<tr>
<td>Good</td>
<td>14.86</td>
<td>3.27</td>
<td>0.926</td>
<td>0.071</td>
</tr>
<tr>
<td>Insane</td>
<td>16.55</td>
<td>3.49</td>
<td>0.913</td>
<td>0.077</td>
</tr>
</tbody>
</table>

**Figure 11:** Baseline response time (RT) per question

**Figure 12:** Response times and accuracy under different distortion profiles
automatic recovery compared to the profile ‘good’. While disproportionately many ‘insane’-distorted questions were skipped in our survey, more than 95% of the time they were answered with an accuracy of approximately 91%. This shows that, albeit difficult to read, message content is generally conveyed even under this distortion profile.

The above usability conclusions are also supported by the qualitative feedback that participants left after completing the survey. Our security results, and hence also our conclusions above, however, should be interpreted in the context of the following three considerations.

Firstly, the arms race between captcha designers and attackers is evolving rapidly. Preprocessing FHEO-generated images using specialised character segmentation logic, as well as the usage of well-tuned machine learning techniques in order to recognise characters individually, have been proven to outperform optical character recognition systems when it comes to breaking captchas [2, 6, 9, 17, 30]. We expect that such techniques will improve the success rate of our adversaries, and hence whenever such attacks become available against FHEO, a re-evaluation becomes necessary.

Secondly, while our security experiments used messages distorted by the same profile, FHEO-generated images that can be harvested from the Internet are distorted under different profiles, and some users take advantage of the custom distortion settings. Hence, our security results do not directly apply to adversaries that operate on images found ‘in the wild’, but must be carefully adjusted in order to take into account the distribution of selected distortion profiles. In order to determine this distribution, we could modify FHEO so that it submits, to the image hosting server, the chosen distortion settings along with the image. Doing so, however, would violate the principle of data minimization, and would therefore require us to collect user consent. Although we currently have no plans for this, such a feature may be added to a future version of FHEO.

Thirdly, according to the legislation of some countries, such as the Digital Millennium Copyright Act [1], Germany’s penalty code §202a), and the United Kingdom’s Computer Misuse Act, automatically recovering the text from FHEO-generated images that are found in the wild, potentially carries a certain legal risk. In light of this, the exact level of protection that FHEO offers against recovery attacks may be, to some extent, irrelevant in the affected territories. That is, even the ‘easy’ distortion profile may suffice to prevent some of the automatic processing of messages protected by FHEO that would otherwise be automatically processed in these territories. We furthermore hope that protecting online messages with FHEO will prove to be a legally effective way to revoke any previously granted consent to all forms of automatic processing that would otherwise apply to these messages.

Finally, note that it is easy for a website to block FHEO-generated images if they are hosted on a well-known set of servers. We believe, however, that reputable organisations are unlikely to do this because the damage to their reputation is likely to outweigh the benefits of blocking.

9. CONCLUSIONS

We presented FHEO and provided an analysis of its security and usability. Our results provide some insights into the usability/security tradeoffs that the different distortion profiles achieve with respect to current, off-the-shelf optical character recognition technology and short online messages. The ‘fair’ and ‘good’ distortion profiles, in particular, achieve a level of protection that seems to justify their respective usability degradation in many situations. The ‘strongest’ distortion profile, on the other hand, does not seem to provide a large additional security margin against the adversaries we considered.

Deciphering distorted text remains an arms race between computers and humans which, on the technical level, is perhaps bound to be won by the former, eventually. We believe that it is, nevertheless, worth pursuing systems that exploit the human advantage for two reasons. Firstly, we expect that the race will still go on for some time, since distortion and attack designers will react to each other’s advances. Secondly, even if it becomes practical to remove the protection offered by FHEO, doing so may remain an unacceptable practice for reputable organisations.

In the future, we intend to improve FHEO’s distortion types by increasing the number of anti-segmentation techniques, as explained in Section 4. We also aim to cooperate with captcha attack designers in order to extend their software so that it can cope with FHEO-generated images. The purpose of this exercise is to evaluate FHEO with dedicated, captcha-breaking software.

10. ACKNOWLEDGMENTS

We would like to thank Ero Balsa, Filipe Beato, Elie Bursztein, Iulia Ion, Roxanna Morote, Stefan Schiffsner, Koen Simoens, and the anonymous reviewers for their insightful comments and suggestions, as well as everyone who participated in our survey. This work was supported by the Flemish Government, IWT SBO SPION, FWO G.0360.11N Location Privacy, FWO G.068611N Data mining, IWT SBO MobCom (a Mobile Companion), IWT GBO SEC SODA, and by the Research Council K.U. Leuven: GOA TENSE. The work was also supported by the European Commission through the support action with contract number 258630 (GINI SA), the CIP thematic network with contract number 270901 (SSEDIC), and the collaborative project with contract number 284862 (FIDELITY).

11. REFERENCES


