Heuristics for the Evaluation of Captchas on Smartphones

Gerardo Reynaga
Carleton University
Ottawa, ON
Canada
gerardor@scs.carleton.ca

Sonia Chiasson
Carleton University
Ottawa, ON
Canada
chiasson@scs.carleton.ca

Paul C. van Oorschot
Carleton University
Ottawa, ON
Canada
paulv@scs.carleton.ca

ABSTRACT

Captchas are used as a security mechanism on the web to distinguish human users from automated programs. However, existing captchas are not well-adapted to mobile devices and may lead users to abandon tasks. Although Web developers have many available captchas, they lack the tools to evaluate if these captchas are suitable for their mobile site. In this paper, we present domain specific usability heuristics for evaluating captchas on smartphones. To assess effectiveness, we compared our proposed heuristics against Nielsen's during evaluations of four captcha schemes on smartphones. The custom heuristics revealed more major problems and more detailed feedback on the problems than Nielsen's.

CCS Concepts

•**Human-centered computing** → **Heuristic evaluations;** *Ubiquitous and mobile computing design and evaluation methods;* •**Security and privacy** → *Web application security;*

Keywords

Captchas, Evaluation, HCI, Heuristic evaluation, HIPs, Smartphones

1. INTRODUCTION

58% of the US population owns a smartphone [39] and saturation is even higher in other places such as the Hong Kong and the UK [36]. In the first quarter of 2013, 84% of US mobile users had used their devices for shopping [37]. Facilitating mobile web interactions is clearly important, as is enforcing web security. In particular, we are looking at one aspect of secure web interactions: captchas. While extensive research is available on captchas for traditional computing [56, 10, 16], work on captchas for mobile devices is limited and more recent [32]. Our recent work has uncovered several usability problems of desktop captchas deployed on websites aimed at smartphones usage [41, 42]. *Captchas* are challenge-response tests used as a security mechanism on the web to distinguish human users from automated programs to reduce malicious use of resources. Differences in form factor, context of use, and I/O modalities between traditional computer and mobile devices suggest that a different approach to captchas may be needed.

Furthermore, the constant attack and scrutiny of captchas is inciting the development of schemes with robust challenges. However, designing a human friendly scheme is also important. Most literature on the usability of captchas pertains to character-recognition (CR) challenges and how distortions should be designed [10, 16, 56]. Nonetheless, until recently, little was published regarding the captcha scheme as an overall piece of software, on the usability of other captcha categories, or captcha on smartphones. Recent captcha schemes are proposing to address the usability issues on mobile devices [46, 28] and our heuristics may help independently evaluate their usability.

Our goal is to develop and validate custom heuristics for evaluating captchas on mobile devices, in particular smartphones. Heuristic Evaluation (HE) [35] provides quick results, avoids costs related to user studies, and yields results useful as either formative or summative evaluations. Any proposed heuristics should be independent of the evaluated software, so we demonstrate their use on four schemes. Most mobile sites still use regular desktop captchas. Evaluating the usability of existing and new captcha schemes should be required before deployment by captcha designers or Webmasters.

Our contributions are: we present our new Mobile Captcha (MC) heuristics, and evaluate their effectiveness. Two heuristic evaluations were conducted on four captcha schemes. The results show that the number and severity of relevant problems found using the MC heuristics is higher than with Nielsen and Molich's general heuristics.

2. RELATED WORK

Expert evaluation techniques, such as HE [35], allow quick and easy usability assessments. Given that a captcha is a relatively simple piece of software, a heuristic evaluation gives evaluators the freedom to explore the scheme from several perspectives. Research in mobile computing and usability has sparked interest in expert evaluations [3, 40]. Since being originally proposed as a methodology by Nielsen, several authors have adapted the original set of heuristics [35] to better fit the requirements of specific application domains. Most relevant to our work, Bertini *et al.* developed heuristics to evaluate usability issues in general applications for *mobile computing*, with emphasis on contextual usage [3].

British HCI 2015, July 13 - 17, 2015, Lincoln, United Kingdom

ISBN 978-1-4503-3643-7/15/07…$15.00

DOI: http://dx.doi.org/10.1145/2783446.2783583

Jaferian *et al.* developed a set of heuristics for IT *security management* tools evaluation. Their heuristics focuses on the collaborative nature of security tools (*e.g.,* tools for threat and vulnerability management) [23]. Other examples include Sutcliffe *et al.* proposed a set of heuristics for *virtual environments* considering natural engagement, natural expression of action, and realistic feedback [49].

Our literature review revealed four common and prevalent methods for developing special purpose usability heuristics. Table 1 lists these methods for domain specific heuristics with examples. The methods are not mutually exclusive; for example, the first and third methods are often combined.

Although orthogonal to our research, accessibility cannot be overlooked. Captchas should be accessible to users with different abilities. For example, audio captcha schemes are meant to provide alternatives for visually impaired users. However, audio captchas on desktop computers have poor usability [4, 45, 10].

3. DEVELOPING THE MC HEURISTICS

Our approach to developing the MC heuristics was to extend and adapt Nielsen's heuristics [35]. Traditional captchas adopted by websites typically become too cumbersome for mobile users [41, 42, 46, 28] and these issues are not easily captured by existing heuristics.

The focus of our work is providing: a) heuristics that find problems hindering correct and efficient human challenge-solving of different captcha schemes on smartphones; and b) a tool for IT practitioners in charge of deploying captchas to quickly and inexpensively assess the usability of available captcha schemes. No usability technique can evaluate the *security* of a captcha scheme and separate methods exists to evaluate security [9, 17, 54]. Even if security is not our focus, there is an implicit assumption that to be viable, captchas must meet at least some minimal threshold of security.

We used the following steps for developing the MC heuristics. *1)* We reviewed the academic literature and identified approximately 50 of the most relevant papers in HCI, mobile HCI, and security. *2)* We summarized documentation on interaction paradigms, input methods, mobile device interaction design, captcha usability, captcha security considerations, mobile internationalization and cultural concerns, and contextual usage. We looked at design guidelines and recommendations from HCI, interaction with mobile devices, and security. *3)* We developed an initial set of heuristics and corresponding guiding questions. We invited expert evaluators with experience on HCI and Security to use the heuristics to evaluate four captcha schemes[1] on smartphones. *4)* After this we refined, combined, and discarded a heuristic that was not capturing or properly reflecting issues and problems. For example, the security heuristic was eliminated since the evaluations were purely based on perception and personal assessment, rather than systematic study. *5)* More HCI evaluators carried out a second heuristic evaluation. *6)* The heuristics went through a second refinement iteration. On this iteration, no heuristics were eliminated or added but some wording and guiding questions were improved. The guiding questions went through multiple iterations between experts in usability and security. During the iterations we turned to the literature to ensure the guiding questions accurately represent the related literature.

[1]Details regarding the evaluation are given in §5.

4. OUR MOBILE CAPTCHA HEURISTICS (MC)

Our proposed heuristics, guiding questions to help evaluators identify problems, their *raison d'être*, and examples of the issues each heuristic evaluates are described below. Heuristics MC1 - MC6 address usability while MC7 focuses on deployability.

MC 1. *User control and freedom.* Input mechanisms required to answer the captcha should not lead users to make mistakes. The scheme should provide controls to correct, retype, or clear the input before submitting. Does the input mechanism obstruct the challenge? Can the user easily obtain a new challenge? Are appropriate controls provided (e.g., zoom, audio pause)? **Rationale**. This heuristic is similar to Nielsen's *User control and freedom* heuristic; while Nielsen's heuristics focuses on providing a way out from an unwanted state, our heuristic considers input artifacts. These artifacts often interfere with the rest of the displayed page. We generalize control mechanisms beyond typed input, *e.g.,* voice input. A related heuristic from the gaming mobile domain is *Device UI and game UI are used for their own purposes*, by Korhonen and Koivisto [25]. Based on direct manipulation, an object should remain visible while the user performs physical actions on it, and the result of these actions should be visible as well. Maintaining visibility of objects and their affordances is challenging on small screens. Others have recognized the small screen as a usability problem leading to typing mistakes [24]. Lentz [27] recommends providing all navigation inside the web application rather than using the browser's navigation controls and providing adequately-sized targets. Designs must also consider that mobile devices could be handled with one or two hands, while standing, sitting, or walking. Typing is the most popular captcha input mechanism [8]. Non-traditional CR schemes may require the user to select by pressing on objects, to trace lines, or to use speech input. Schemes should also provide means to deselect, correct or erase user actions.

MC 2. *Learnability.* The scheme should provide, and not require more than, brief instructions. The captcha scheme should be intuitive, without excessive cognitive load. Can the user figure out how to use the captcha quickly/easily? Is there any guidance? Can a small set of instructions explain it sufficiently? Are such instructions provided? Is the scheme understandable? **Rationale**. This heuristic derives from Nielsen's *Help and documentation.* In addition, *Learnability* is a usability goal [43]. Ideally, the scheme can be used with minimal or no documentation, especially since the captcha challenge is an obstacle to their overall objective. Korhonen and Koivisto [25] suggest related heuristics: *the player does not have to memorize things unnecessarily*, *the game contains help*, and *the game provides clear goals or supports player created goals.* Similarly, Carroll's [11] minimalist approach to instruction presents only material essential to performing the task and uses an action-oriented approach. Mobile best practices and design guidelines advocate for clear and simple language, labelling all form controls, properly positioning the labels in relation to their form controls, and making it simple for users to access help [53, 12]. The growing number of captcha schemes and their increased adoption may lead to user confusion. A common learnability issue with text captchas is that users are unsure whether they must respect the case of the challenge. The

Approach to developing domain specific heuristics	Domain examples	Heuristics
1. Adapt or extend Nielsen's	Ambient Displays [31]	12
	Security [57]	6
	Virtual Reality [49]	12
	Mobile Computing [3]	8
2. Base on theory from target domain	Security [23]	7
	Computer Supported Collaborative Work [19]	5
3. Transform design guidelines	Mobile [3]	8
	Web services [50]	7
	General advice [43]	(advice)
4. Use domain experience & analysis of usability problems	General [35]	9
	General [34]	10

Table 1: Prevalent methods for developing domain specific evaluation heuristics.

basic captcha task should simple and intuitive but this is rarely the case. For instance, many audio schemes require users to recall long strings or multiple words.

MC 3. *Efficiency of use.* The scheme should be quick and efficient to use. The scheme should avoid controls or non-captcha related artifacts that may lead to errors or inefficiencies, and minimize unnecessary complexity. Do any barriers hinder solvability of the challenge? Is the captcha solvable within a reasonable time? **Rationale**. Nielsen's original *Flexibility and efficiency of use* heuristic focuses on allowing customizations for frequent and advanced users. In contrast, our heuristic addresses issues that may create stumbling blocks that hinder solvability and decrease the efficiency of the captcha scheme. Bertini *et al.* [3] refer to efficiency of use as allowing mobile users to dynamically configure the system based on contextual needs. In the MC heuristics *efficiency* refers to the way a captcha scheme supports users in completing the challenge: if needed, provide control buttons, allow users to obtain new challenges, and if security is not negatively impacted, include play controls (*e.g.,* pause and play controls for audio schemes). For example, a challenge that requires swapping alpha and numeric keyboards further decreases efficiency. Solving captchas on smartphones may become tedious. In related work, researchers found that scrolling, zooming and panning, and screen size contribute to loss of context, all of which can negatively affect captchas [47, 2, 7].

MC 4. *Input mechanisms.* The scheme should support input mechanisms allowing easy completion of the challenge on different devices. Does the input mechanism cause any additional problems for completing the challenge? Does the scheme have appropriate input methods for the target device? Does the scheme allow device appropriate input alternatives? Does it allow for user input preference? **Rationale**. This heuristic derives from Bertini's *Ease of input, screen readability and glanceability* which has similar reasoning to ours: "Mobile systems should provide easy ways to input data, possibly reducing or avoiding the need for the user to use both hands" [3]. Ten years ago there were only two paradigms for mobile input: pen and keyboard input [30]. Now, we also find touch-sensitive screens, speech input, gyroscopes, accelerometers, and other input mechanisms. Moreover, there are a number of mobile text input options, including the unistroke letter set, optimized on-screen keyboard layouts, and soft keyboard error correction. Furthermore, users have to switch their attention between the keyboard and the input area due to the lack of tactile feedback, making this activity error-prone [30]. Consider the case in which the device allows for voice input, but the captcha challenge is in a another language. The speech recognition mechanism may not recognize the answer, or the user is burdened with changing language settings. Similarly CR schemes in non-roman alphabets could require special text entry solutions [29].

MC 5. *Solvability.* Captcha challenges should be simple for users to solve while preventing automated computer solutions. Challenges should minimize user confusion and be suitable for mobile devices (with small screens, and restricted input methods). Does the challenge contain ambiguous elements that could lead to confusion for users? Do the distractors make it too difficult to solve the challenge? **Rationale**. This new heuristic represents a delicate balance between the secure and the usable. Captcha schemes employ various distortion or distraction techniques to prevent adversarial attacks. However, excessive or poorly designed distractions can lead to challenges with unacceptably low solution rates. Challenges, regardless of the class, should be easy for humans, hard for machines [9, 56, 44]. Security design guidelines for CR captchas suggest using multiple fonts, sans-serif fonts, and varied font sizes [9, 56]. If not carefully applied, these design guidelines can have a negative impact on solvability. From the mobile perspective, security features may not fit device constraints. For instance, if a CR challenge image is too large, it may not be rendered by the device [53]. If the image is too small, distorted and complex, the user may need to zoom and lose overview [7]. Some audio challenges are implemented as spoken characters and may be prone to confusion (*e.g.,* `a/8`) [44]. Another consideration for audio captchas is that audio distortion techniques are commonly used to prevent attacks; however, a noisy environment may make the challenge unsolvable. Content-confusion is not unique to audio challenges; CR challenges may also include confusable characters (*e.g.,* `5/S/s,nn/m`).

MC 6. *User perception.* Captchas should be pleasant to solve and should cause no discomfort to users. Does the scheme have potential to cause discomfort (*e.g.,* eye strain or nausea) Is the scheme pleasant to use? Is the scheme usable and acceptable? **Rationale**. Nielsen's closest heuristic is *Aesthetic and minimalist design*, which pertains to dialogs. Korhonen and Koivisto's [25] mobile gaming heuristic: *the players are rewarded and rewards are meaningful* is linked to gamers' satisfaction and engagement. Our *User*

perception pertains to the user's subjective satisfaction during use. While users typically cannot choose the type of captcha, they can choose whether to continue using the site serving it. A mobile user's expectations and experience is impacted more easily due to the variable context of use [13]. Hiltunen *et al.* [21] and Botha *et al.* [5] identify the following factors influencing the mobile experience: user (unique attributes of mobile users, cultural context, and skills), interaction techniques, task context (multitasking, interruptible and mobile), physical context, social context, technological context, privacy and security, device, and connection (network and bandwidth concerns). Solving captcha challenges on smartphones can be frustrating if the scheme is too challenging, or if the captcha requires more cognitive load than the user is willing to dedicate to the task. Yan and El Ahmaad [56] also include *satisfaction* as one of their usability criteria for captchas.

MC 7. *Consistency with user's localization and environment.* The scheme should be universal[2] and should be suitable for the range of situations and environments in which users may access the scheme. Language and culture should not impose additional barriers to solving challenges. Are the task demands appropriate for the environments in which users will be accessing them? Is the challenge independent of culture, language, location? Does the scheme provide ample time to solve the challenge (*i.e.,* time-out does not impede solvability)? **Rationale**. Nielsen's heuristics do not explicitly address localization and environment, but the *Consistency and standards* heuristic relates to following platform conventions. Other heuristics related to contextual usage are *the game accommodates with the surroundings* and *interruptions are handled reasonably* [25]. Localization "refers to the adaptation of a product, application or document content to meet the language, cultural and other requirements of a specific target market (a locale)." [52]. From the perspective of mobile devices Cui and Roto [15] identify four contextual factors: spatial (mobile or stationary), temporal (duration of breaks), social (alone or in group) and access factors (WLAN or cellular network). Lee *et al.* [26] classify mobile Internet use into personal and environmental contexts. Personal context pertains to the state of the users themselves in terms of emotion, time and movement. Environmental context include all factors external to the user: location, lighting conditions, distraction, and crowding (*e.g.,* other people, social interactions) in the immediate environment. In terms of captchas and their deployability, a captcha scheme is often chosen based solely on the reasoning that it is a popular solution. Although most captcha design suggestions include universality, not all deployed schemes achieve this goal. Challenges should be independent of users' physical location, language, or culture [44]. Whether the challenge is friendly to non-native speakers is another usability concern [56]. As part of captcha deployment, IT practitioners must also consider that captcha schemes will be accessed from a variety of devices. Captcha schemes should be portable to various devices, screen sizes, and input mechanisms.

[2]Universal design is the term used to reflect a particular perspective upon the design of interactive products and services that respects and values the dimensions of diversity intrinsic in human capabilities, technological environments and contexts of use [48].

5. ASSESSING THE MC HEURISTICS

The most common approach to validate heuristics is to evaluate an application with both the proposed heuristics and Nielsen's heuristics [35]. This approach varies from proposal to proposal in the thoroughness of the analysis. For instance, Jaferian *et al.* apply the two sets of heuristics to an identity management system [23]. Baker *et al.* [1] assess heuristics to evaluate groupware. In both cases, two main factors are analyzed and compared against Nielsen's: a) problems found (false positives, raw, consolidated), and b) inspector performance (average performance, consistency of individual performance, proportion of inspectors who found each problem, ranking of individual performance, number of inspectors required to uncover a good number of problems). The ultimate criterion for assessing the effectiveness of usability evaluation methods is finding real usability problems [20]. Hartson define realness as: "a usability problem is real if it is a predictor of a problem that users will encounter in real work-context usage and that will have an impact on usability (user performance, productivity and/or satisfaction)" [20]. Specific issues related to character-recognition captchas are addressed in previous work [55, 54, 8]. The first evaluation consisted of using the proposed heuristics in a standard heuristic evaluation process assessing four captcha schemes (§6). In the second evaluation we repeated the process with Nielsen's & Molich's heuristics for comparison.

Target Schemes. The schemes were chosen because they represent the main captcha categories: character-recognition (CR), image-recognition (IR), and moving-image object recognition (MIOR). The target schemes are depicted in Figure 1. *reCaptcha* [18] is a widely deployed on the Internet; this CR challenge consisting of recognizing and typing two words. *Asirra*[3][33] is a research IR captcha; the challenge consisting of identifying images of cats and dogs. *NuCaptcha* [38] is a commercial MIOR scheme consisting of reading alphanumeric characters that overlap as they swing independently left to right (statically pinned at the centre of each letter). *Animated* captcha, (Vappic 4D) [51], is an experimental captcha MIOR consisting of six alphanumeric characters arranged in a patterned cylinder that rotates in the centre of the captcha screen. The similarly patterned background portrays the base where the cylinder sits; this floor swivels up and down. We used live demo sites offered by the scheme owners to test the latest version of the schemes. The focus of evaluation is the heuristics and is not intended as endorsement of any specific captcha scheme.

We also pilot tested audio schemes and found these are unusable on smartphones due to their high operational complexity and strong need for recall. Mobile-specific captchas were unavailable at the time of the evaluation because they were still prototypes.

6. THE MC HEURISTICS EVALUATION

A heuristic evaluation (HE) was done by experts using the MC heuristics to examine the four target captcha schemes. Our evaluation method followed Nielsen's [35] recommendations; with a few modifications, described below. Since captchas are not feature rich programs, our HE design allowed evaluators to create their own task scenarios using the questions listed in §4 as a guide. The general task was to solve challenges and explore the overall interface.

[3]The Asirra captcha service was closed in October 2014.

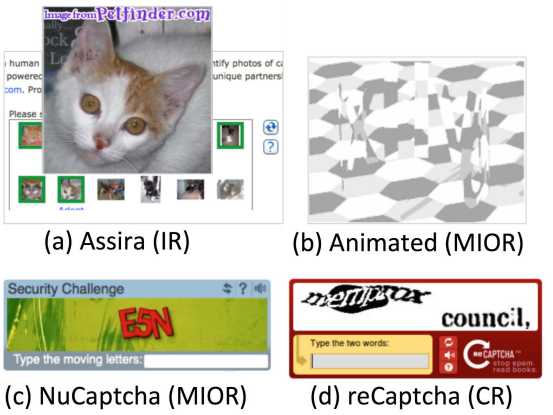


Figure 1: Target Schemes. Images are from the respective websites: (a)[33], (b)[51], (c)[38], (d)[18].

Evaluators: To recruit evaluators, we e-mailed known experts in HCI who also were familiar with Security. Once the evaluators agreed, we e-mailed details on how to conduct the heuristic evaluation. Nine evaluators (4 females, 5 males) completed a median of 4 schemes each. Evaluators rated their knowledge of security and HCI; their means were 4.1 and 4.2 respectively (out of 5, N=9). Evaluators included people in the mobile industry, academic researchers with proven expertise, and students with at least one graduate course in security and HCI. ***Equipment***: Participants completed the evaluations on their own smartphones and in the environment of their choosing. There was one Nexus S, one Galaxy Nexus, 2 iPhones 4, 4 iPhone 4s, and one Samsung Focus (SGH-i917). While evaluating a scheme, evaluators used a secondary device to compile feedback from the heuristics. ***Procedure***: Prior to the evaluation, evaluators responded to a demographics questionnaire. A host web page presented links to the four captchas hosted on third party demo sites. Thus, experts evaluated and experienced live versions of the captcha schemes. Each evaluator conducted his/her assessment independently. For each scheme visited on the evaluator's smartphone, they assessed its merits based on the heuristics, noted in Limesurvey[4] any problems uncovered, rated each problem's severity, and provided an overall score based on compliance with the heuristic.

6.1 Data Analysis

We summarized the usability problems identified by each evaluator and then generated an aggregate list of problems per scheme. We used a variant of Thematic Analysis [6] to process, synthesize and categorize the reported problems. The categorization was conducted by two researchers. The steps were: *1)* Problem synthesis (*open coding*): We separated and synthesized the *raw problems*. Raw problems include problem descriptions with compound issues which can be decomposed with better granularity. Compound issues refer to more than one action, or describe more than one problem, or are where ambiguous problem descriptions can be better synthesized. *2)* Consolidate problems (*axial coding* and *selective coding*): We consolidated overlapping problems and identified *false alarms*. Consolidation started with an empty list; a problem was added to if it did not yet exist in the list. Otherwise, a problem-frequency counter was updated. In heuristic evaluations, it is not unusual for two evaluators to write two completely different descriptions for the same problem. Consistent with the literature, we define *overlapping* problems as those identified by two or more evaluators. A *false alarm* was identified as a problem that "was not verified by any of the researchers" [22], or when "the reasoning of the evaluator in describing the problem was fallacious" [23]. The result is a list of *unique* problems for each scheme. *3)* Tagging the problems with heuristics (*defining and naming themes*): two researchers reviewed each unique problem to verifying whether it fits in the assigned heuristic. Sometimes evaluators identified problems under a particular heuristic, but these problems fit better under a different heuristic; we refer to these as *misclassifications*. These misclassifications were re-tagged. Re-tagging was done with mutual consensus. We categorized the unique problems into major (2) and minor (1). *Major* problems would significantly hinder usability, prevent the user from solving the challenge, or lead to mistakes. The original rating by evaluators was considered when classifying. All evaluators were asked to review the four schemes, but not every evaluator completed all schemes. Eight evaluators completed reCaptcha, while six assessed Asirra, NuCaptcha and Animated. We asked them to spend one hour in total, but the evaluation may have taken longer. Evaluators could stop when tired.

6.2 Results

Raw Problems. We analyzed raw problems to assess the evaluators' consistency in finding problems and assigning severity ratings. The total number of raw problems per scheme across heuristics is as follows: reCaptcha (86), Asirra (60), NuCaptcha (31), Animated (79). Evaluators found problems relating to *MC3: Efficiency* and *MC4: Input Mechanisms* most severe and those that are related to *MC7: User Location* least likely to significantly impact usability. This is likely because most schemes require zooming and panning to answer the challenge thus affecting efficiency. Evaluators also found that restrictions on input mechanisms considerably hindered usability. The two heuristics that most promote problem discovery are *MC1: User control* and *MC3: Efficiency of use*, with 53 and 43 problems across the four schemes. *MC7: User's localization* and *MC2: Learnability* led to the least discovery, with 31 and 23 problems described respectively. Based on severity ratings for *MC5: Solvability*, Asirra appeared most solvable and Animated had the most critical solvability problems.

Unique Problems. The number of unique problems are more representative of real issues, thus for the purpose of evaluating the heuristics, better conclusions can be obtained from this refined set. Table 2 summarizes the number of unique problems per heuristic. Fewer unique problems indicate that the scheme performed better with respect to the given heuristic. The heuristics with the most unique problems were *MC1: User control* and *MC2: Learnability*, with 25 and 22, respectively. Fewer unique problems were discovered by *MC4: Input mechanisms* (15) and *MC1: Localization* (17). reCaptcha had the most unique problems for *MC5: Solvability*, however, its mean severity rating was 2.1 indicating a large number of relatively minor issues. This highlights the importance of including severity ratings in evaluations. The number of unique problems for *MC3: Ef-*

[4]LimeSurvey web based survey tool http://www.limesurvey.org/

	Synthesized Unique Problems								Evaluators
Scheme	MC1	MC2	MC3	MC4	MC5	MC6	MC7	Total	
reCaptcha	6	9	8	6	8	3	3	43	8
Asirra	11	4	5	3	5	6	3	37	6
NuCaptcha	3	5	0	3	2	4	1	18	6
Animated	5	4	7	3	6	10	4	39	6
Total	25	22	20	15	18	20	17	137	

Table 2: MC Heuristics. Unique problems found by heuristic per scheme.

ficiency is quite variable; reCaptcha gathered the most efficiency problems while NuCaptcha had none.

Evaluator Performance. Regardless of the heuristics, evaluators perform differently from each other. HCI background, domain expertise, time, incentives, and experience all have an influence. The distribution of unique problems across evaluators per scheme is depicted in Figures 2. In addition to using Nielsen's original depiction [35], our graphs use colours to show the severity rating assigned by evaluators. Stronger evaluators are at the bottom of the graph since they found the most problems. The definition of strong-vs-weak evaluators is related to the evaluator's ability to find problems. The problems are also ordered, from left (easy) to right (hard). An *easy* problem is found by multiple evaluators; it is an "easy" problem to identify, versus a hard problem that is rarely found. The characterization of easy-vs-hard problem and strong-vs-weak evaluator are common interpretations introduced by Nielsen [35]. We noticed that some easy problems are overlooked by strong evaluators, while some hard problems are only found by strong evaluators. In addition, the distribution shown by the schemes is consistent with that of Nielsen's and existing literature [35, 23]. The colouring of the graphs helps to visually identify the overall severity of problems per scheme. For example, the Animated scheme may have fewer unique problems (39) compared to reCaptcha (43), however the darker colours of the graph shows that evaluators found more severe problems for Animated. The colours also highlight the degree to which evaluators agree, or disagree, on the severity of any given problem. It may also show patterns within each expert's own rating of problems. That is, does expert A assigns consistently high or low severity ratings to the problems they report? Within each scheme, evaluators tend to assign a higher severity to the 'easiest' problems identified. Indicating that the heuristics are helpful in guiding evaluators at uncovering more severe problems.

Overall Rating per Heuristic. Evaluators provided an overall Likert scale rating (1=significant problems to 10=excellent) of each scheme per heuristic. These ratings provided an overall perspective of the problems discovered for each heuristic and their severity. The ratings further provided supporting evidence for the analysis of the unique problems. *MC1: User control and freedom* helped evaluators identify significant issues with most schemes. *MC2: Learnability* uncovered mostly minor issues. Although three schemes used the virtual keyboard as input mechanism, not all schemes scored equally; *MC4: Input mechanisms* highlighted these differences. *MC6: Perception* underlined the importance of testing innovative captchas; one evaluator noted "these captchas made me nauseous" (Animated).

7. NIELSEN HEURISTIC EVALUATION

A comparison HE was done using Nielsen and Molich's heuristics. ***Evaluators***: We recruited from the LinkedIn SIGCHI - SIG on Computer-Human Interaction. We also recruited HCI graduate students. The same methodology as for the MC heuristics was followed (§6). We limited participation to nine evaluators (3 females, 6 males); these completed a median of 4 schemes each. Evaluators rated their knowledge of security and HCI, their means were 3.44 in both (out of 5, N = 9). There was one Nexus 5, one Samsung Nexus, two LG Nexus 4, one iPhone 5S, one iPod, one Windows Phone, and two Galaxy S III. ***Procedure***: We followed the same procedure as described in §6.

7.1 Data Analysis

Analysis followed the same procedure as in §6.1. The categorization was also conducted by two researchers. We provide brief results here; comparisons between MC heuristics and Nielsen's are described in §8.

7.2 Results

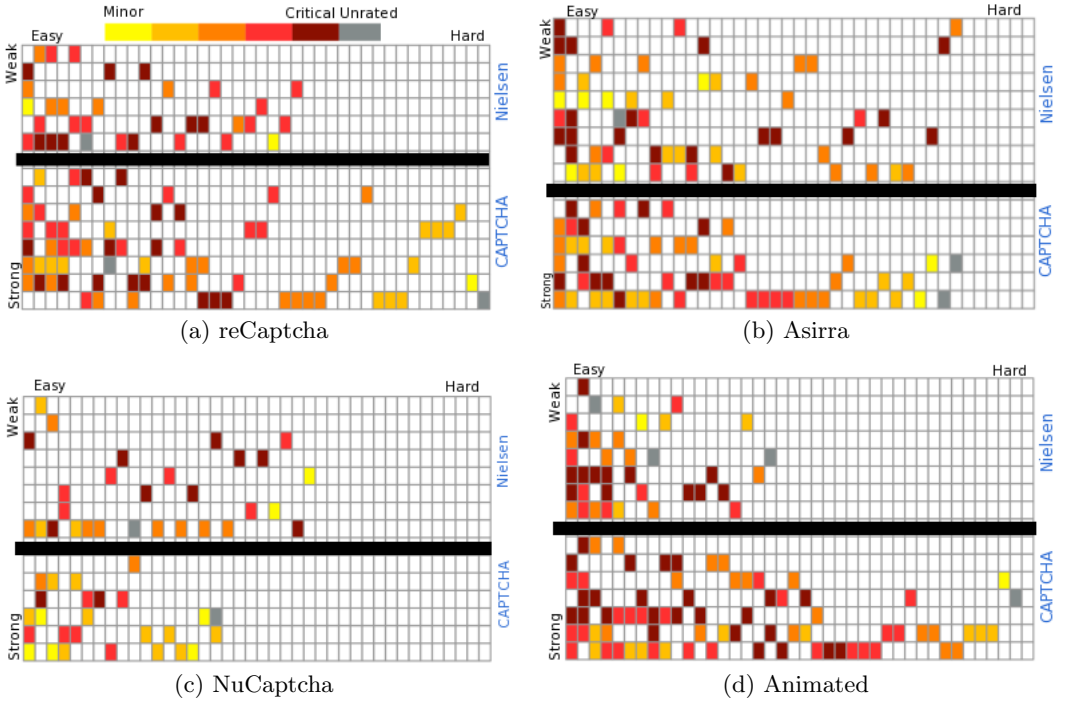
Raw Problems. The number of raw problems per scheme across is as follow: reCaptcha (36), Asirra (73), NuCaptcha (37), Animated (46). Similarly to MC heuristics, evaluators described many problems in the first heuristic, *Visibility of System Status* (44); we speculate evaluators use this heuristic as starting point and noted problem here even if it was not the best match. *Error Prevention* also promoted a high number of problem descriptions (37).

Unique Problems. Nielsen's heuristics with most unique problems were *Help and documentation* (21) and *Visibility of System Status* (10) respectively. Fewest unique problems were classified in *Aesthetic and minimalist design* (4), *Consistency and standards* (3), *Help users recognize diagnose, and recover from errors* (3). The number of false positives was: reCaptcha (6), Asirra (13), NuCaptcha (10), and Animated (5).

Evaluator Performance. The distribution of unique problems across evaluators per scheme is depicted in Figures 2. Nielsen's unique problems are more scattered (*i.e.,* less agreement between evaluators) than MC heuristics. This may be due to the lack of specificity of Nielsen's heuristics for this domain. As initially concluded by Nielsen, the original set of heuristics do not necessarily cover all domains [35].

8. EFFECTIVENESS OF THE MC HEURISTICS

Table 3 shows the classification of problems for both. Naturally, there are similarities between the two since our set has heuristics derived from Nielsen's. However, MC heuristics facilitated uncovering more unique problems than Nielsen's despite the fact that Nielsen's HE had more evaluators per



(a) reCaptcha (b) Asirra

(c) NuCaptcha (d) Animated

Figure 2: Unique problems per captcha scheme. Each row corresponds to an evaluator and each column to a problem. Nielsen's heuristics above the black bar and the MC heuristics below; they should be interpreted separately.

scheme. This suggests that our heuristics does not require a large number of evaluators to uncover a similar number of problems. The number of *false positives* was lower with MC heuristics and more major problems were identified. We attribute this to the customization of our MC heuristics for this domain. Regarding the number of evaluators, we intended the evaluation of the four schemes to be independent from each other. Nielsen and Molich [35] recommended 3 to 5 evaluators which we exceeded.

According to existing literature, it is desirable for heuristic evaluations to be *thorough*, *valid*, and *consistent* (for reliability) [20]. We compared our results along these previously defined metrics [20, 14]. The norm is to visually compare these metrics [20, 14]. The ultimate criterion to assess usability evaluation methods is finding real usability problems, §5. We define the set of real problems from the unique problems derived from the MC HE and our earlier user study evaluating these same schemes [41]. To test effectiveness, we explore the following question: How well do the MC heuristics help evaluators discover real problems? Our effectiveness metrics are based on the synthesized unique problems from the two heuristic evaluations.

8.1 Comparison between MC and Nielsen

Thoroughness: The heuristics should find as many usability problems as possible. Thoroughness indicates the proportion of problems found by the heuristic evaluation to real problems [20, 14].

$$Thoroughness = \frac{Number\ of\ real\ problems\ identified}{Number\ of\ real\ problems\ that\ exist}$$

Thoroughness ranges from 0 to 1, with higher values indicating more thoroughness. The MC heuristics score over 70% on all schemes. Other published domain heuristics [23, 14] have shown 71% and 55% thoroughness, respectively. We note that MC heuristics are consistently thorough across schemes. Nielsen's thoroughness values have considerable variability across schemes.

Validity: Validity measures how well the heuristic serves its intended purpose; or the proportion of problems found that are real usability problems [20, 14]. The *Number of issues identified as problems* includes the number of *False positives* identified plus the *Number of real problems identified* (Table 3).

$$Validity = \frac{Number\ of\ real\ problems\ identified}{Number\ of\ issues\ identified\ as\ problems}$$

Validity ranges from 0 to 1. The heuristics perform consistently with a high number of real problems identified and a small number of false positives, yielding high validity. Nielsen's HE showed more false positives. High Validity scores indicate that most of the problems are real, thus higher values are better. In comparison, Chattratichart and Lindgaard [14] obtained a validity of 0.65.

Reliability: Consistent results are important independent of the individual performing the heuristic evaluation [20]. It is also desirable to identify major usability issues [23]. Reliability measures the mean number of evaluators finding a problem [14]. Our reliability scores are similar to the 0.178 reported by Chattratichart and Lindgaard [14]; lower scores are better. Nielsen's HE had more evaluators but was less reliable across schemes.

Scheme	Unique		False P.		Major		Minor		Raw		Thorough.		Validity		Reliability	
	MC	N	MC	N	MC	N	MC	N	MC	N	MC	N	MC	N	MC	N
reCaptcha	43	26	5	6	26	9	17	17	86	43	0.79	0.68	0.89	0.81	0.18	0.30
Asirra	37	32	3	13	14	14	23	18	60	82	0.71	0.76	0.92	0.71	0.16	0.28
NuCaptcha	18	27	3	10	6	10	12	17	31	46	0.70	0.84	0.91	0.73	0.28	0.33
Animated	39	18	2	5	24	7	15	11	79	52	0.73	0.54	0.95	0.78	0.15	0.44
All	137	103	13	34	70	40	67	63	256	223						

Table 3: Heuristic Evaluations. Note. MC = **Captcha heuristics,** N = **Nielsen's heuristics.**

$$Reliability = \frac{Number\,of\,evaluators}{Number\,of\,real\,problems\,identified}$$

Effectiveness Summary. The *Thoroughness*, *Validity* and *Reliability* results align with previous work using the same metrics to determine effectiveness. The metrics further support the effectiveness of the MC heuristics.

9. CONCLUDING REMARKS

The MC heuristics performed well in finding usability problems. Compared to Nielsen's, MC found more problems with higher severity. As suggested by Nielsen and Molich [35], our domain specific heuristics performed better at uncovering problems than their general heuristics. Scheme-based differences in severity ratings, unique problems and overall performance also indicate that the heuristics performed well. Identical results for all evaluated schemes would have indicated flawed heuristics since the schemes are representatives of different captcha categories.

So far the heuristics were used by people with well founded knowledge of HCI and security. We plan to ask captcha developers and web administrators to use the MC heuristics in an evaluation to verify the usefulness for this target group. Additional analysis looking at the evaluators' background may be useful in some circumstances; for example, when HE evaluators do not share a homogenous background. Exploration based on evaluator's background could include similarities or differences on the severity and types of issues reported, and the likelihood of finding easy problems or discovering hard problems. Heuristics specifically considering audio schemes could be added to the present set of heuristics. The MC heuristics could be used to evaluate captchas developed for desktops, however we think that they could be further optimized for this context.

Evaluating the security of any captcha scheme is a non trivial task which requires significant expertise. The approach to "breaking" a scheme varies according to the captcha class. For example, assessing the security of character recognition captchas requires a different set of tools than an image recognition captcha. A heuristic evaluation cannot be designed to evaluate the security of any piece of software. Thus, the MC heuristics are meant to be used in parallel with a security analysis.

Our aim was to propose and validate a set of heuristics for evaluating captcha schemes on smartphones. To validate the heuristics, we conducted two heuristic evaluations. The MC heuristics can be used as a discount method to uncover usability problems in captchas meant to be used in sites targeting mobile devices. Furthermore, the heuristics can used at any stage of the design of new captcha schemes for mobile devices.

10. ACKNOWLEDGMENTS

Sonia Chiasson holds a Canada Research Chair in Human Oriented Computer Security, and Paul van Oorschot holds a Canada Research Chair in Authentication and Computer Security. Both authors acknowledge the Natural Sciences and Engineering Research Council of Canada (NSERC) for funding their Chairs and Discovery Grants. The authors also acknowledge funding from NSERC ISSNet and SurfNet.

11. REFERENCES

[1] K. Baker, S. Greenberg, and C. Gutwin. Empirical development of a heuristic evaluation methodology for shared workspace groupware. CSCW, pages 96–105. ACM, 2002.

[2] J. Bergman and J. Vainio. Interacting with the flow. MobileHCI, pages 249–252. ACM, 2010.

[3] E. Bertini, S. Gabrielli, and S. Kimani. Appropriating and assessing heuristics for mobile computing. AVI, pages 119–126. ACM, 2006.

[4] J. P. Bigham and A. C. Cavender. Evaluating existing audio captchas and an interface optimized for non-visual use. CHI '09, pages 1829–1838, New York, NY, USA, 2009. ACM.

[5] A. Botha, M. Herselman, and D. van Greunen. Mobile user experience in a mlearning environment. SAICSIT, pages 29–38. ACM, 2010.

[6] V. Braun and V. Clarke. *Thematic analysis*, pages 57–71. APA handbook of research methods in psychology, Vol 2: Research designs: Quantitative, qualitative, neuropsychological, and biological. American Psychological Association, 2012.

[7] S. Burigat, L. Chittaro, and E. Parlato. Map, diagram, and web page navigation on mobile devices: the effectiveness of zoomable user interfaces with overviews. MobileHCI, pages 147–156. ACM, 2008.

[8] E. Bursztein, J. Aigrain, A. Moscicki, and J. C. Mitchell. The end is nigh: Generic solving of text-based captchas. WOOT. USENIX Association, 2014.

[9] E. Bursztein, R. Beauxis, H. Paskov, D. Perito, C. Fabry, and J. Mitchell. The failure of noise-based non-continuous audio captchas. SSP, pages 19 –31. IEEE, May 2011.

[10] E. Bursztein, S. Bethard, C. Fabry, J. C. Mitchell, and D. Jurafsky. How good are humans at solving CAPTCHAs? A large scale evaluation. SSP, pages 399–413. IEEE, 2010.

[11] J. Carroll and H. van der Meij. Ten misconceptions about minimalism. *IEEE Trans. on Prof. Comm.*, 39(2):72 –86, jun 1996.

[12] L. Cerejo. The elements of the mobile user experience.

http://mobile.smashingmagazine.com/2012/07/12/elements-mobile-user-experience/, 2012.

[13] A. Charland and B. LeRoux. Mobile application development: Web vs. native. *Queue*, 9(4):20:20–20:28, Apr. 2011.

[14] J. Chattratichart and G. Lindgaard. A comparative evaluation of heuristic-based usability inspection methods. CHI, pages 2213–2220. ACM, 2008.

[15] Y. Cui and V. Roto. How people use the web on mobile devices. WWW, pages 905–914, New York, NY, USA, 2008. ACM.

[16] A. El Ahmad, J. Yan, and W. Ng. Captcha design: colour, usability and security. *Internet Computing, IEEE*, 16(2):44–51, 2012.

[17] P. Golle. Machine learning attacks against the Asirra CAPTCHA. CCS, pages 535–542. ACM, 2008.

[18] Google. reCaptcha: Stop Spam, Read Books. http://www.google.com/recaptcha, 2013. Last visited: Feb 2015.

[19] S. Greenberg, G. Fitzpatrick, C. Gutwin, and S. M. Kaplan. Adapting the locales framework for heuristic evaluation of groupware. *Australasian Jnl of Information Syst.*, 7(2), 2000.

[20] H. R. Hartson, T. S. Andre, and R. C. Williges. Criteria for evaluating usability evaluation methods. *Int. Journal of Human-Computer Interaction*, 13(4):373–410, 2001.

[21] M. Hiltunen, M. Laukka, and J. Luomala. *Mobile user experience*. IT press, 2002.

[22] E. T. Hvannberg, E. L.-C. Law, and M. K. Lárusdóttir. Heuristic evaluation: Comparing ways of finding and reporting usability problems. *Interacting with Computers*, 19(2):225–240, 2007.

[23] P. Jaferian, K. Hawkey, A. Sotirakopoulos, M. Velez-Rojas, and K. Beznosov. Heuristics for evaluating it security management tools. SOUPS, pages 7:1–7:20. ACM, 2011.

[24] J. Kjeldskov. "Just-in-Place" information for mobile device interfaces. *LNCS*, 2411:271–275, 2002.

[25] H. Korhonen and E. M. I. Koivisto. Playability heuristics for mobile games. MobileHCI, pages 9–16. ACM, 2006.

[26] I. Lee, J. Kim, and J. Kim. Use contexts for the mobile internet: A longitudinal study monitoring actual use of mobile internet services. *Int. J. of HCI*, 18(3):269–292, 2005.

[27] J. L. Lentz. User interface design for the mobile web. http://www.ibm.com/developerworks/web/library/wa-interface/index.html, 2011. Last visited: October 2014.

[28] C.-J. Liao, C.-J. Yang, J.-T. Yang, H.-Y. Hsu, and J.-W. Liu. A game and accelerometer-based captcha scheme for mobile learning system. WCEMHT, pages 1385–1390. AACE, 2013.

[29] Y. Liu and K.-J. Räihä. Rotatxt: Chinese pinyin input with a rotator. MobileHCI, pages 225–233. ACM, 2008.

[30] I. MacKenzie and R. Soukoreff. Text entry for mobile computing: Models and methods, theory and practice. *Int. J. HCI*, 17(2-3):147–198, 2002.

[31] J. Mankoff, A. K. Dey, G. Hsieh, J. A. Kientz, S. Lederer, and M. Ames. Heuristic evaluation of ambient displays. In *CHI*, pages 169–176. ACM, 2003.

[32] H. Meutzner, S. Gupta, and D. Kolossa. Constructing secure audio captchas by exploiting differences between humans and machines. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 2335–2338, New York, NY, USA, 2015. ACM.

[33] Microsoft. Asirra. http://research.microsoft.com/en-us/um/redmond/projects/asirra/, 2011. Last visited: Oct 2014.

[34] J. Nielsen. Ten usability heuristics. http://www.nngroup.com/articles/ten-usability-heuristics/, 2012. Last visited: Jun 2014.

[35] J. Nielsen and R. Molich. Heuristic evaluation of user interfaces. CHI, pages 249–256. ACM, 1990.

[36] The asian mobile consumer decoded. http://www.nielsen.com/us/en/newswire/2013/the-asian-mobile-consumer-decoded0.html, 2013. Last visited: Jan 2014.

[37] Who is the mobile shopper? http://www.nielsen.com/us/en/newswire/2013/who-is-the-mobile-shopper-.html, 2013. Last visited: Jan 2014.

[38] NuCaptcha. White paper: Nucaptcha and traditional captcha. http://www.nucaptcha.com/resources/whitepapers, 2014. Last visited: Feb 2015.

[39] C. Pew Research. Mobile Technology Fact Sheet. http://www.pewinternet.org/fact-sheets/mobile-technology-fact-sheet/, 2014. Last visited: Feb 2015.

[40] S. Po, S. Howard, F. Vetere, and M. B. Skov. Heuristic evaluation and mobile usability: Bridging the realism gap. *LNCS*, 3160:49–60, 2004.

[41] G. Reynaga and S. Chiasson. The usability of CAPTCHAs on smartphones. SECRYPT, pages 427–434. SCITEPRESS, August 2013.

[42] G. Reynaga, S. Chiasson, and P. C. van Oorscho. Exploring the usability of captchas on smartphones: Comparisons and recommendations. In *Workshop on Usable Security USEC*. Internet Society, February 2015.

[43] Y. Rogers, H. Sharp, and J. Preece. *Interaction Design: beyond human-computer interaction, 3rd Edition*. John Wiley & Sons, 2011.

[44] Y. Rui and Z. Liu. Artifacial: Automated reverse turing test using facial features. *Multimedia Systems*, 9:493–502, 2004.

[45] G. Sauer, J. Holman, J. Lazar, H. Hochheiser, and J. Feng. Accessible privacy and security: a universally usable human-interaction proof tool. *Universal Access in the Information Society*, 9:239–248, 2010.

[46] S. Shirali-Shahreza, G. Penn, R. Balakrishnan, and Y. Ganjali. Seesay and hearsay captcha for mobile interaction. CHI, pages 2147–2156. ACM, 2013.

[47] S. Shrestha. Mobile web browsing: usability study. Mobility, pages 187–194. ACM, 2007.

[48] C. Stephanidis and D. Akoumianakis. Universal design: towards universal access in the information society. CHI EA, pages 499–500. ACM, 2001.

[49] A. G. Sutcliffe and B. Gault. Heuristic evaluation of

virtual reality applications. *Interacting with Computers*, 16(4):831–849, 2004.

[50] K. Vaananen-Vainio-Mattila and M. Waljas. Evaluating user experience of cross-platform web services with a heuristic evaluation method. *Int. J. of Arts and Tech.*, 3:402–421, Oct. 05 2010.

[51] Vappic. 4d captcha. http://www.vappic.com/moreplease, 2012. Last visited: Jun 2014.

[52] W3C. Localization vs. internationalization. http://www.w3.org/International/questions/qa-i18n, 2012. Last visited: Oct 2014.

[53] W3C. Mobile web best practices 1.0. http://www.w3.org/TR/mobile-bp/, 2012. Last visited: Mar 2014.

[54] Y. Xu, G. Reynaga, S. Chiasson, J.-M. Frahm, F. Monrose, and P. C. van Oorschot. Security analysis and related usability of motion-based captchas: Decoding codewords in motion. *IEEE TDSC*, 11(5):480–493, 2014.

[55] J. Yan and A. El Ahmad. Breaking visual CAPTCHAs with naive pattern recognition algorithms. ACSAC, pages 279–291, Dec. 2007.

[56] J. Yan and A. S. El Ahmad. Usability of CAPTCHAs or usability issues in CAPTCHA design. SOUPS, pages 44–52. ACM, 2008.

[57] A. Zhou, J. Blustein, and N. Zincir-Heywood. Improving intrusion detection systems through heuristic evaluation. volume 3, pages 1641 – 1644, May 2004.