Carnegie Mellon University

CARNEGIE INSTITUTE OF TECHNOLOGY

THESIS

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF Doctor of Philosophy

TITLE Improving Computer Security Dialogs: An Exploration of Attention And Habituation

PRESENTED BY Cristian Antonio Bravo Lillo

ACCEPTED BY THE DEPARTMENT OF

Engineering and Public Policy

Lorrie Cranor — February 25, 2014
ADVISOR, MAJOR PROFESSOR — DATE

M. Granger Morgan — February 25, 2014
DEPARTMENT HEAD — DATE

APPROVED BY THE COLLEGE COUNCIL

Vijayakumar Bhagavatula — February 28, 2014
DEAN — DATE

Improving Computer Security Dialogs: An Exploration of Attention and Habituation

Submitted in partial fulfillment of the requirements for
the degree of
Doctor of Philosophy
in
Engineering and Public Policy

Cristian Bravo-Lillo
B.S., Computing Engineering, Universidad de Chile

Carnegie Mellon University
Pittsburgh, PA
May, 2014

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346

Keywords: usable security, computer security dialogs, attractors, habituation, attention.

Abstract

Computer dialogs communicate important security messages, but their excessive use has produced habituation: a strong tendency by computer users to ignore security dialogs. Unlike physical warnings, whose design and use is regulated by law and based on years of research, computer security dialogs are often designed in an arbitrary manner. We need scientific solutions to produce dialogs that users will heed and understand.

Currently, we lack an understanding of the factors that drive users' attention to security dialogs, and how to counteract habituation. Studying computer security behavior is difficult because a) users are more likely to expose themselves to risk in a lab experiment than in daily life, b) the size of observed effects is usually very small, which makes it necessary to collect many observations, and c) it is complex to balance research interests and the ethical duty not to harm.

My thesis makes two contributions: a novel methodology to study behavioral responses to security dialogs in a realistic, ethical way with high levels of ecological validity, and a novel technique to increase and retain attention to security dialogs, even in the presence of habituation.

To Almendra,
who in June 6, 2007 at 5:19 am CLT was 2.6 parsecs from us
and had an elevation of -23° 54' 07" as seen from S 33° 27' 36", W 70° 38' 24",
but then rose to be the brightest star in the sky (besides the usual celestial bodies),

and to Verónica,
who is not a star but rises every night in the western sky,
never further away than 47° from the Sun,
to be the brightest object in the sky (besides the usual celestial bodies).

Acknowledgments

First and foremost, thanks to my committee: Lorrie Cranor, Julie Downs, Marvin Sirbu, and Stuart Schechter. This research would certainly not have been possible without your guidance and encouragement.

I am deeply grateful to Lorrie for the right dosage of sharp criticism, compassionate support, academic rigor and great patience and wisdom as my advisor. Somehow every time I split verbs I remember that poetic license hanging on your office, and I quickly hurry to my dictionary and thesaurus, to no avail. I am not sure that I ever expressed how grateful I am for all your help and dedication throughout these years. Thanks, truly! I am also deeply indebted to Stuart, who besides befriending and supporting me, is the true intellectual father of some key ideas in my thesis. Your very kind invitation to Microsoft Research changed my perspective when I needed it the most. I deeply enjoyed my time in Seattle, and every moment with you ever since! Lorrie and Stuart, you are both brilliant researchers that I can only aspire to follow. Working with you has been an enriching and humbling experience, and your guidance has made of me a better researcher. I am truly looking forward to future collaborations with you both. I am also very grateful to Julie for her great insights and lots of help with statistics and experimental design; you taught me more than you are probably aware of.

I am also indebted to my co-authors: Saranga Komanduri, Rob Reeder, Manya Sleeper, as well as Lorrie, Julie, and Stuart; and to current and past peers at the wonderful CUPS lab research team: Rebecca Balebako, Alain Forget, Patrick Kelley, Pedro León, Aleecia McDonald, Rich Shay, Blase Ur, and Kami Vaniea. I am especially thankful to Saranga Komanduri for all the help, and encouragement, and all the conversations about philosophy, gaming, religion, family, technology, R, experimental design and life. You are an amazing researcher, and a wonderful friend! I am grateful to Pedro León and his wonderful family for supporting Almendra, Verónica and I in a way that is hardly understandable for someone who is not latino. I wish we remain friends for a long time to come. ¡Muchas gracias, Pedro!

Thanks to all those who helped us with our research (mentioned in alphabetical order): John Douceur (Microsoft Research), Serge Egelman (UC Berkeley), Stephen Fienberg (Carnegie Mellon), Cormac Herley (Microsoft Research), Mandy Holbrook (Carnegie Mellon), David Molnar (Microsoft Research), Greg Norcie (Indiana), Adam Shostack (Microsoft), Rick Wash (Michigan State), and all the anonymous reviewers of our papers for their helpful reviews and suggestions.

Lastly, I thank my wife, Verónica, and my parents, Verónica and Manuel. Without you all, I would quite literally, and at so many different levels of meaning, not be here. This achievement, if any at all, goes to you, as you deserve it more than I do.

This research was funded in part by National Science Foundation grants CNS083-1428, CNS1116934, and DGE0903659. Part of this research was conducted when the author was an intern at Microsoft Research.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In 2012, 81% of the US population had access to Internet, and there were about 310 million mobile telephone subscriptions – about 98 subscriptions per 100 inhabitants [44]. Despite 86% of U.S. Internet users having taken measures to avoid being scammed or tracked, 21% of Internet users have had a social network or an email account hijacked, and 11% have suffered the online theft of important information such as credit card, bank account, or social security numbers [69]. Although estimating the aggregated economic harm is a highly debated issue [4, 41, 38], a study in 2012 estimated the harm produced worldwide by online payment card fraud to be around $4.2 billion in direct costs and $30 billion in indirect costs, per year [4] [1].

Web browsers play a critical role in protecting people from these and other types of socially engineered scams, not only on desktop and laptop computers but also on smartphones and tablets. Although it would be ideal to completely automate all security decisions that involve potential harm to computer users, it is currently infeasible or impractical to take the human out of the loop [24, 31]. Thus, web browsers (as well as operating systems and productivity software, like email clients and word processors) should communicate security decisions to humans in a faithful, accurate, and timely fashion.

Computer Security Dialogs are small windows that interrupt a user's primary task to communicate that a security decision needs to be made. A computer security dialog is a type of warning that is used in computer systems, and as such, it shares many of the benefits and problems of physical warnings. Despite their importance, computer security dialogs are routinely ignored by humans: most people do not read security dialogs [34, 85, 87], do not understand them [29, 7], or simply choose not heed them [64, 78], even when the situation is clearly hazardous. Security dialogs are also considered an annoying interruption regardless of factors like dialog size and level of engagement in the primary task [6]. Despite these problems, software designers often rely on users to perform important security tasks, including judging whether or not to heed a security dialog [24].

This seemingly irrational behavior – ignoring security advice that may help computer users to avoid risk – is mainly due to overexposure. Security dialogs have been overused by software developers in operating systems, productivity applications, and web browsers. As a consequence,

[1] The authors did not specify a currency in their study, since "currency isn't important given the accuracy of the figures available to us." The authors explicitly state that their numbers should be considered only as an order-of-magnitude estimation.

computer users have been overexposed to security dialogs, which in turn has produced habituation [18, 24, 34, 66, 74, 85, 87]. Authors have reported specific aspects of habituation that began to occur after just a few exposures to a new dialog [18], or after a single exposure to a dialog that resembled other dialogs that participants had seen previously [34]. A second reason for ignoring security dialogs is poor design: many computer security dialogs are poorly written, and are full of technical jargon [13, 74].

Some authors argue that given these problems, the rejection of security advice is actually very rational [38, 41]. Yet, it is still desirous for habituated and reluctant computer users to pay attention to security advice in those cases in which the risk is significant and a decision cannot be taken automatically on behalf of users.

Current literature on human attention to warnings in general, and to security dialogs in particular, is abundant and growing (see the discussion in Section 2.1.4.) However, we lack specific, quantitative knowledge about the factors that drive people's attention to security dialogs [19]. Similarly, we do not know how quick habituation builds up, whether we can counteract its effects, and if so, how to effectively do so.

Studying security behavior is difficult for at least two reasons. One is related to ecological validity; the other one pertains to statistical effect size.

First, participants are often well aware of their participation in a research study, and adopt self-selected roles that seek to provide the answers they think the researchers want. Similarly, being in a lab environment makes them feel reassured; they behave in a risk-prone way that usually differs from how they would behave in a real-world situation. Sotirakopoulos et al. provided "evidence of a strong bias of the laboratory environment for usable security studies" that may have motivated the one third of their participants mentioned to behave differently than if they were not in the lab when responding to SSL certificate dialogs [85]. Krol et al. reported that 30% of participants spontaneously mentioned "being in the lab" as a reason for ignoring a security dialog that appeared when downloading a PDF file [51]. Furthermore, participants tend to focus on finishing the task that was assigned to them, and in doing so they might miss or skip important security cues. Schechter et al. showed that participants playing a role are less likely to respond to clues that indicate the presence of risk than those who believe they are actually at risk [78]. Krol et al. reported that 20% of participants in their experiment ignored a dialog because they wanted to "get to the task" [51].

Second, when a researcher attempts to isolate one of the multiple factors that influence computer users' response to security dialogs, any changes introduced in a dialog should be small. For example, one should change either an icon, or the text of a button, but not both; otherwise, the observed effect cannot be attributed to any of the individual changes. In addition, small changes in a dialog usually produce small effects. To illustrate, imagine a researcher who adds the word 'Recommended' next to an option in a dialog, to measure whether this change increases the proportion of users who click on the corresponding option. Exposing each participant to more than one dialog would introduce learning effects and increase the chance that participants might guess the purpose of the study. A between-subjects design with one data point per participant becomes inevitable. Some participants will not notice the factor that is being studied. Out of those who notice it, some will not think it is important. Out of those participants who consider the message important, some will not understand it. Out of those who understand it, some may think that the recommendation is simply not applicable to their case, and so on [97]. Thus, many observations

have to be collected in order to balance statistical significance (α) and power ($1 - \beta$). However, bringing many participants to the lab is not cost-effective, and is subject to the problems described above.

In order to tackle these problems, I first designed a method that allowed me to observe the security behavior of many computer users, in a cost- and time-effective manner, with high levels of ecological validity. Then, I applied the method to determine which of a number of factors influence computer users' behavior. I designed a novel technique – *attractors* – to increase participants' attention to important security information contained in computer dialogs. Finally, I applied the invented method to test both the effectiveness of attractors, and their resilience to heavy, repeated exposure.

1.1 Research questions

> *In this thesis, I make two contributions: a novel methodology to study behavioral responses to security dialogs in a realistic, ethical way with high levels of ecological validity, and a novel technique to increase and retain people's attention to security dialogs, even in the presence of habituation.*

I present the results of one lab experiment and nine online experiments in this thesis. My primary motivation in conducting these experiments was to gain a better understanding of both attention and habituation to security dialogs. These studies were conducted to answer the following questions:

1. Do the ideas and preconceptions that inexperienced computer users hold negatively affect their responses to security scenarios presented to them through computer security dialogs? (Chapter 3)

2. Does increased and richer information provided through security dialogs help inexperienced users to take qualitatively better security decisions? (Chapter 4)

3. How can we observe computer users' behavior responding to realistic security tasks in a controlled environment that does not expose them to unnecessary risk, and does not systematically bias them into risk-averse or risk-prone behavior? (Chapter 5)

4. Can we modify security dialogs to increase users' attention to the information contained in the dialog's context-dependent fields? Do these interface modifications improve user response to security dialogs in a way that a) increases the overall security of the system, and b) does not decrease the usability of the system? (Chapter 6)

5. Do these interface modifications improve users' response in a way that is resilient to repeated exposure to dialogs? (Chapter 7)

6. Does user attention to dialogs decrease when the amount of text presented in a dialog increases? Do users who believe they have antivirus software installed respond to security dialogs in a more risk-prone way? (Chapter 8)

1.2 Thesis overview

In Chapter 2, I briefly review part of the abundant literature in warnings and human attention. I start by describing computer security dialogs and their most important features. Next, I provide evidence of how participants in multiple studies ignore security dialogs in different stages of the exposure process. Then, I describe three theoretical concepts that help to explain why humans ignore security dialogs. Finally, I describe some of the early approaches in usable security research to direct and retain computer users' attention to security dialogs, as antecedents to my research.

In Chapter 3, I describe a lab experiment aimed at understanding how novice and experienced users differ in their responses to security scenarios, presented to them through computer security dialogs. This exploratory study provided my co-authors and I with a number of possible lines of reasoning for novice and expert users, allowed us to identify how these lines differ, and suggested ways in which we could leverage experts' experience and knowledge into creating better security dialogs that could help novice users to make more informed decisions.

In Chapter 4, I report on a follow-up, online experiment designed to apply the knowledge that my co-authors and I gained from the previous study. We picked four computer dialogs and redesigned them using the insights we gained previously, as well as the input from three interface designers with experience in software development. We recruited participants from Amazon's Mechanical Turk service, and asked them to respond to role-playing scenarios that included an image of either the original or a redesigned computer dialog. Overall, we observed that participants tended to take fewer risks when presented with the redesigned dialogs, but we did not observe an increase in participants' understanding of the presented situation. This study also highlighted many of the limitations that one may incur when using self-reported data.

Motivated by the limitations of the previous experiments, and by the need to observe participants' performance in realistic security tasks that neither bias them nor expose them to unnecessary risk, I designed and implemented a set of online tasks that I used in most subsequent studies. In these tasks, users are instructed to play three online games on three different websites, and to answer a few questions about each game. The first two games came from pre-existing gaming websites that were outside of researchers' control. The third was a simulated gaming website that displayed a fake security dialog to which users had to respond. In this last website, all of the participants' actions were carefully recorded: mouse clicks, movements and timing. After users responded to this prompt, they were given an exit survey and were debriefed about the true nature of the experiment.

Our purpose in designing this method was to strive for ecological validity while not increasing the risk that participants would be exposed to if they participated in a traditional lab study. In Chapter 5, I describe the method in detail. I used this method to determine how many users would enter their credentials in simulated password-entry dialogs, both in the MS Windows and the Mac OS operating systems. This was a known but previously unquantified problem. I designed a set of security dialogs that required the entry of user passwords to install simulated software, and measured how many of these entered credentials would correspond to real user credentials. I found with 95% of confidence that at least 15% of users would be compromised by such a deception, and at most 38% of participants would actually notice the deception. In addition to these findings, this experiment forced my co-authors and I to carefully consider the ethical implications of deceiving users in online experiments.

In Chapter 6, I turn to the problem of driving computer users' attention to information contained in security dialogs. I describe a novel approach that I designed and successfully used to decrease participants' risky, unconscious behavior due to lack of attention to risk signals presented in security dialogs. I describe two experiments that compared users' response to security dialogs; some of these dialogs included interface modifications – *attractors* – designed to make participants aware of the information presented in the dialogs. Overall, I found that users who saw these attractors significantly decreased their unaware, risky behaviors compared to those participants who did not see any attractors.

One of the main problems of the previous approach is its novelty. Participants who had never before seen an attractor may have paid attention to the dialog only because they were unfamiliar with these odd-looking interfaces. In Chapter 7, I describe two experiments. The first one was aimed at determining if these attractors were effective after heavy, repeated exposure. We found that most attractors were still effective after participants clicked repeatedly for 2.5 minutes in dialogs included these attractors. The second experiment was designed to test whether attractors not only performed well under repeated exposure, but also whether their effectiveness remained constant with increasing levels of habituation. We found that two out of five attractors performed satisfactorily under these conditions.

In Chapter 8, I describe two experiments aimed at understanding whether a number of specific factors may affect participants' attention to dialogs. We found evidence that in a suspicious scenario, both increasing the length of the text in a dialog (up to a point) and putting important information in the end of a dialog decrease the proportion of correct responses to tested dialogs.

In Chapter 9, I describe three existing incentives for software vendors in the U.S. to display more security dialogs than they should; I explain why an excessive number of dialogs is harmful, and I offer suggestions of policy actions that may help to bring down the overall number of security dialogs presented.

Finally, in Chapter 10, I present a summary of the findings of previous chapters, a list of recommendations to software developers and dialog designers aimed at designing more effective security dialogs, and I describe some intended future work.

Chapter 2

Background and Related work

In this chapter, I summarize a small fraction of the abundant literature that is relevant to the subjects under study. In the background section I describe the purpose and scope of warnings, and their origination in the United States. Then, I explore some important features of computer security dialogs, a specific type of warning used in computers. Next, I describe briefly a model of human attention to warnings – the human-in-the-loop framework. This model will be useful to understand how warnings fail in obtaining computer users' attention, and to understand why the solution I propose in Chapter 6 works. Finally, I describe what we know about habituation, a difficult problem that we will attempt to solve in Chapter 7.

The second section reports evidence found in literature about how and why people ignore computer security dialogs, emphasizing the kind of failure in attention that was observed based on the human-in-the-loop framework. Finally, I describe related work about the trusted path problem – a specific issue that arises when trust is required in the relationship between humans and interfaces, e.g., when entering a web banking system. Reviewing this problem is important because it is closely related to the one I am trying to solve: how to drive computer users' attention to trusted security information on the screen.

2.1 Background

Computer Security Dialogs (or security dialogs for short) are small pop-up windows that are displayed on a computer or smartphone screen, interrupting the user to present a security decision to be made. Usually, the type of decisions that are presented to computer users are about whether to obtain a new functionality (perhaps by installing software from a third party) or engage in a new task that offers some level of risk to the user. A computer security dialog always offers at least two options to the user. One of them represents a safe course of action, which will not expose the user to any risks but will not offer the benefits of the new functionality. The other option will provide the aforementioned benefit, exposing the user to some risks. Additional options may be offered with variations of the previous alternatives. Three examples of computer security dialogs are shown in Figures 2.1, 2.2, and 2.3.

Computer Security Dialogs are a specific type of warnings, used in computer systems. As such, many properties of warnings are also met by security dialogs. Throughout this thesis, I use the term 'warning' to refer almost exclusively to physical warning communications, like the ones

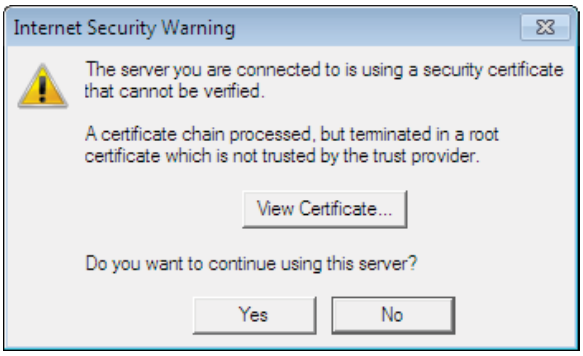


Figure 2.1: Example of a Computer Security Dialog shown by Microsoft Office Outlook 2013. This dialog is shown when a connection is established to a server, and the server sends back an SSL certificate that is either self-signed, or is signed by an authority that has not been acknowledged as trustworthy by the user.

we may find in pharmaceutical or industrial products[1]. In contrast, I will use the term 'security dialogs' to refer exclusively to the small windows described in the previous paragraph.

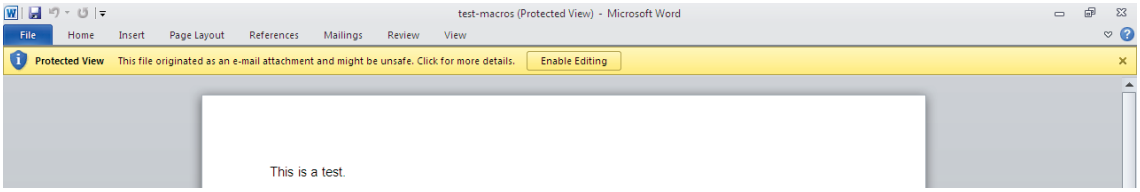


Figure 2.2: Example of Computer Security Dialog shown by Microsoft Office Word 2010. This dialog is presented on top of a Word document (yellow bar) when a document is downloaded from the Internet, and has macros embedded that makes the document possibly unsafe to open. The user is not allowed to edit the document unless the button 'Enable editing' is clicked.

In the rest of this section, I describe first the purpose, scope, and history of physical warnings (from Section 2.1.1 on), which constitute a superset of computer security dialogs. Next, I describe some specific characteristics of computer security dialogs (Section 2.1.3). Finally, I review what we know about human attention to warnings and security dialogs (Section 2.1.4), and about habituation to security dialogs (Section 2.1.5).

2.1.1 Purpose and scope of warnings

Warnings are communications designed to protect people from harm: they may seek to modify people's behavior, to promote compliance with safety regulations, "to reduce or prevent health problems, workplace accidents, personal injury, and property damage," to remind already-aware people of a hazard, and, as a legal instrument, to transfer liability from the maker of a product to the consumer [98]. A warning can include a variety of components that affect its effectiveness at achieving these goals, including its visual design (e.g., size, colors, graphics), use of "signal words," length, and interactivity [53].

[1]The one exception to this convention is the term 'phishing warnings', which is broadly used to refer to computer security dialogs that warn the user about a possible phishing website.

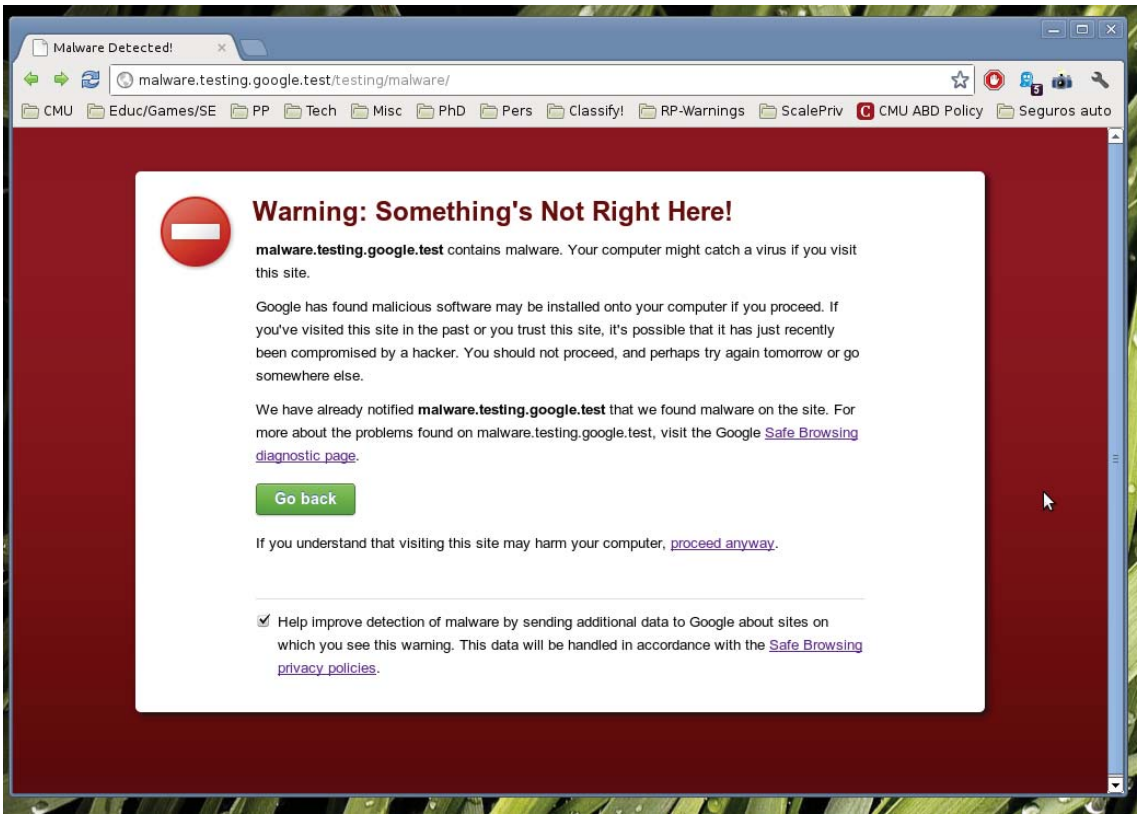


Figure 2.3: Example of a Computer Security Dialog shown within Google's Chrome browser. This dialog is shown whenever a user reaches a website that has been included in Google's Safe Browsing blacklist [40].

An effective physical warning will clearly communicate the risk, the consequences of not complying, and instructions on how to comply (although some of this information can be omitted if the risk is obvious or the consequences may be deduced from the warning) [98]. Many of the most common computer warnings fail to follow one or more of these guidelines. For example, in the warning dialog shown in Figure 2.1, there is no explanation of the risk ('You might not be connecting to the server you think you are connecting to'), no explanation of the consequences ('You might be disclosing private information to unknown parties'), and no instructions on how to avoid the risk ('Check the certificate, and verify with an offline source the identity of the owner of the certificate.')

Warnings should also serve as a third-line defense against hazards in a well-accepted hierarchy of actions to be taken, known as the "hazard control hierarchy" [55, 98]. The most common version of the hierarchy states that, if possible, a risk should be eliminated; otherwise, it should be guarded; only when that is not possible, a warning should be used [24, 98].

Consider for example a hazardous broken sidewalk (Figure 2.4). The first and preferred solution should be to repair the sidewalk (i.e., design the risk out). If repairing the sidewalk is either not feasible or not cost-effective, a barricade can be put around it (i.e., guard against the risk). Otherwise, a warning can be put next to the sidewalk. Warnings can be posted as an interim solution,

but they should not be the only safeguard.

In many practical situations it is not feasible to design a hazard out or to guard against it. For example, the sharp edge of a knife cannot be designed out without making the knife useless, and it is not practical to guard against the risk of cutting oneself. Similarly, the risk of being phished by a malicious website cannot be designed out completely, although guarding strategies can be employed, such as automatically detecting and removing suspicious links from email.

2.1.2 Origin of warnings in the United States

Warnings as we know them today were created in the United States during the first decade of the 20th century when employees started to successfully sue their employers for accidents at their workplace. If employers provided clear and explicit warnings about the risks, they could usually avoid lawsuits entirely or decrease their liability [53]. During the 19th century there were virtually no regulations about warnings in the United States. The first warnings were auditory and visual signs aimed at preventing railroad accidents. Warnings in products were rare, and their shape and wording was entirely up to manufacturers. For example, poison manufacturers sold their product in bottles that were shaped as skulls or coffins with a ridged surface. Although some of these bottles also contained the word "Poison", both the shape and color of the bottle served as reminders of the nature of the product for children and illiterates [35].

With the industrial growth of the United States in the first half of the 20th century, workers started to work for long hours in harsh conditions. Accidents were frequent, possibly as a result of sleep deprivation and exhaustion. Workers started to sue their employers for work-related accidents. Employers used two types of defense in courts: proving employee negligence, or stating that employees knew about the risks and worked anyway. If employees could prove that they were not negligent, and that employers failed to provide proper warnings about risks in their workplace, they were usually compensated. This greatly increased the interest of industries to produce warnings that could accurately communicate the risks in workplaces [35].

In 1913, the National Safety Council (NSC) was formed as an industry-led effort to promote safety through the standardization of warnings in workplaces. This effort was an attempt to shift responsibility from factories and industry to workers. In 1928, the NSC published a pamphlet named "Warning signs: their use and maintenance" that offered guidance for warning designers that would become common over the next years [35].

In March of 1927, the US Congress passed the Federal Caustic Poison Act, which set standards for labels to be put on bottles containing a number of toxic substances like lye and ammonia. This Act also created the Food, Drug, and Insecticide Administration, which would become the Food and Drug Administration (FDA) in 1930 [35].

During the 1930s new lawsuits regarding cases of silicosis and asbestosis threatened the financial stability of several industries, including insurance companies. Between 1934 and 1952, the Manufacturing Chemists Association (MCA) collaborated with the US Surgeon General to create better warnings for specific chemical products. This effort allowed the greatly affected chemical industry to partially counteract the threats of litigation and regulation. The MCA published a guide named "Manual L-1: A guide to precautionary labeling of hazardous chemicals." Although this guide did not cite any scientific studies, it contained advice that would also became common in following years, like the use of a panel word ("Danger", "Warning" or "Caution") in contrasting colors for warnings [35].

During the years after the Second World War, there was a new wave of industrial growth in the United States that brought with it an increase in the demand for medications and home goods like refrigerators and TVs, which in turn increased the need for warnings [35]. In subsequent years, more restrictive laws would appear both at the federal and state level, mandating warnings for many kinds of products, including the automobile and pharmaceutical industries, and consumer products like alcohol, tobacco and toys.

As a consequence of the manner in which warnings developed in the U.S., the legal theories of negligence and strict liability were developed to allow a plaintiff to sue a company for failure to warn about a hazard. In Chapter 9 I will describe these theories and explore their applicability to software.

2.1.3 Computer Security Dialogs

Warnings are also used in computer systems, usually on the brink of an impending danger to users' information or to their identity credentials [24]. Unlike physical warnings, computer security dialogs are not permanently displayed: they are dynamic dialogs, triggered whenever the conditions set by software developers are met. The content of the dialog is usually decided based on those conditions. In this sense, a computer dialog is a template: part of its content and appearance is fixed, and the remainder corresponds to placeholders that are filled out with information right before displaying it. Throughout this work, I will refer to these templates as *computer security dialogs* and to the placeholders as *context-dependent fields* or *salient fields*. If two dialogs coming from the same template are shown to the user, I will refer to these as two instances of the same dialog.

Exogenous hazards

Unlike warnings placed in pharmaceutical or industrial products, wherein the manufacturer is the source of both the hazard and the warning, many computer security dialogs warn about hazards for which the software vendor is not the source. For example, today most browser vendors include phishing warnings that are based on blacklists – compilations of known phishing websites, e.g., Google' Safe Browsing API [40]. Phishing scams are not perpetrated by browser vendors; the hazard would completely disappear if the criminals that are responsible for these scams stopped perpetrating them. However, browser vendors know about this risk, and ship their products with a phishing warning.

Dependence on the context

Computer security dialogs are necessary in today's applications, in part due to the complexity of the interactions between humans and computer systems. When software is designed and built, developers must try to anticipate the conditions under which software will be used. However, many of these interactions depend on how, when, and by whom software is used. Consider the warning that appears on a bottle of poison that may be accidentally drunk. In this type of warning, the message cannot be interpreted in two ways: this liquid is harmful for your health, do not drink. Computer security dialogs are not like the advice in such a warning, but more like the advice one may find on a bottle of alcohol: do not drink if you are going to drive a car, or if you are pregnant.

Figure 2.4: Broken sidewalk in Pittsburgh, Pennsylvania. Picture courtesy of Lorrie Cranor.

It is not possible for the designers of a warning to know in advance whether the receiver of the warning is about to drive a car or is pregnant. It is up to the receiver to interpret the warning and decide if it is applicable to her.

An example may help to illustrate the point. Imagine a user that receives an email with a file attached. Her email client automatically scans the attachment and finds nothing suspicious. Since new malware threats are constantly being created, it is not safe to assume that the attachment is free from all malware. The user then needs to pay attention to the context. If the sender is unknown to the user, and the message does not contain any personal details addressed to her, it is likely that the attachment is not safe and should not be opened. On the other hand, if the sender is known, the user is expecting an email with an attachment (maybe from a colleague that is sitting next to her), and the content of the message and the user's expectations about it match, then the attachment is likely safe and the user should open it.

Due to this dependence on the context, it is currently impossible to automate all security and privacy decisions, taking the user out of the loop. In some cases it is not even desirable to automate these decisions [31]. Very often security dialogs are the final check to prevent users from infecting their laptops with malware, falling for phishing schemes, leaking their financial or personal information to scammers, or a number of other equally unpleasant or harmful consequences. Security dialogs are thus fundamental to current computer systems, and it is unlikely that we will be able to remove either dialogs from computer systems or humans from security decision-making in the near future. Given this scenario, we should strive to improve security dialogs to help computer users to make more informed security decisions.

Safe response instead of compliance

In warnings literature, response to a warning is often evaluated in terms of *compliance* [17, 46, 61, 98]. Compliance has been defined as performing an action when instructed to do so [61]. We believe that the notion of *safe response* is more helpful when applied to computer security dialogs. In prior work we defined *safe response* as taking the least risky option provided by a computer security dialog [16]. In the context of the example as above, the safe response would be not to open the email attachment, as this is the only response that would present no risk to the user.

Any other response, such as opening or saving the attachment, would present some level of risk. However, if the user is expecting to receive the attachment, knows and trusts the sender, and the content of the email matches what she would expect in that situation, then we might consider this as a low-risk context. In this situation, a safe response is not necessary and she should open the attachment.

High levels of safe response are not always necessary, as has been argued above. A computer security dialog is effective if it helps a user to apply her knowledge of the context to make an informed decision that balances risk and usability. Usually, there is a trade-off between usability and level of risk that is based on the specific context. Always making the least risky choice would allow for a completely safe system but would reduce functionality. We consider safe response to be a desirable response in high-risk contexts, under the assumption that users should protect themselves against the high risk of a potential threat. Similarly, we consider safe response as being an undesirable response in low risk contexts, under the assumption that it is unnecessary for users to block functionality in these situations. Sunshine et al. took a similar approach in their evaluation of user response to web browser certificate dialogs on an online banking login page (high risk) and a university library website (low risk) [87].

2.1.4 The human in the loop

In this section, I describe the two most relevant existing models for human attention. The first one is a general model that comes from psychology literature: the Computer-Human Interaction Model, developed by Wogalter [97]. The second is a more specific model, tailored to computer users' interactions with five types of security messages, the Human-In-The-Loop framework, developed by Cranor [24]. These models provide a theoretical framework that allow us to analyze the different stages that occur whenever a person faces a computer security dialog.

The Computer-Human Information Processing model

The Communication-Human Information Processing (C-HIP) model describes the human processes involved in the internalization of a warning [97]. The model assumes two agents, the *source* and the *receiver*, and describes a set of sequential stages with feedback loops that the receiver should pass through, with flow of information or processing from one stage to the next, until a change of behavior attributable to a warning happens. If the warning is successful, the behavior change will protect the receiver from harm. Each stage represents a necessary condition for the stages that follow. Wogalter [97] describes the different stages of the model:

1. *Attention switch*: The warning captures the receiver's attention. In this phase a warning has to compete for the receiver's attention with other stimuli present in the environment, possibly including other warnings.

2. *Attention maintenance*: The receiver decides to pay extended attention to the warning. Warnings need a certain minimum time span of attention to be decoded and internalized. If too short, the message may not be read in its entirety and might be misunderstood.

3. *Comprehension, Memory*: The receiver understands the warning and commits the message to her working memory.

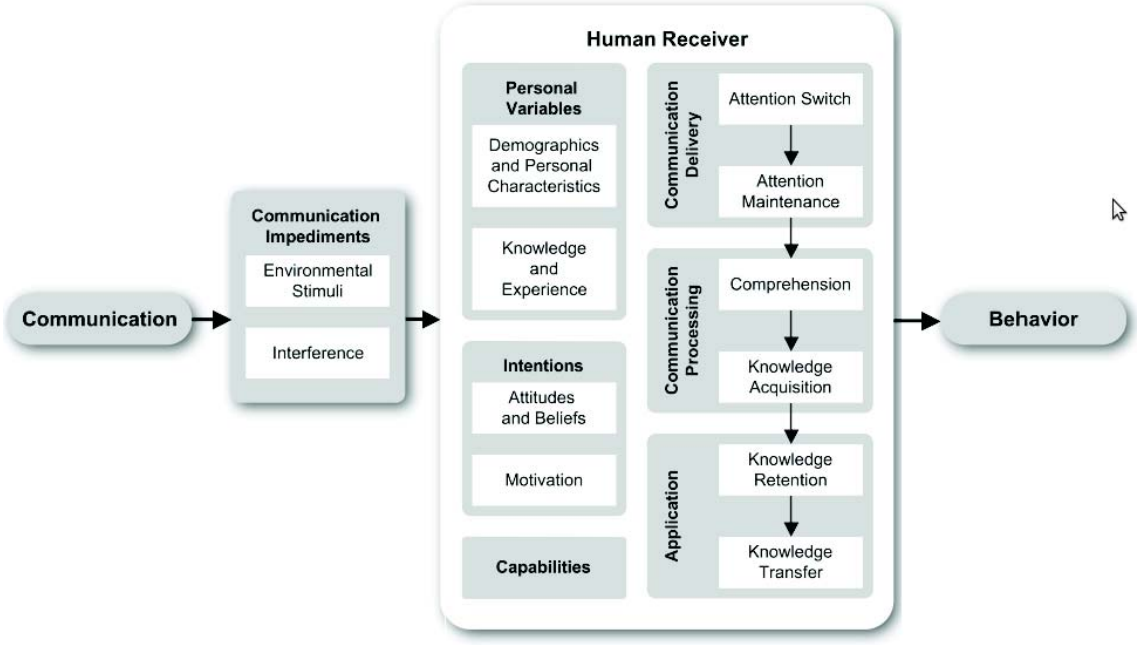


Figure 2.5: The Human-In-The-Loop Framework, proposed by Cranor [4].

4. *Attitudes, Beliefs*: The receiver judges that the warning is applicable to her.

5. *Motivation*: The receiver perceives that it is important to heed the warning.

6. *Behavior*: Finally, the receiver changes her behavior to comply with the warning.

One specific factor that may affect the attention maintenance stage is text length and saliency. If the text is too long, the receiver might decide that it is not worth reading; if the warning is not salient enough, the receiver might decide not to read in the belief that if it were really important it would have been bigger [97, p.55].

In the case of computer dialogs, the source is the software displaying the dialog (e.g., the operating system), and the developers and designers of the application. The receiver is the user of the software.

The Human-In-The-Loop framework

Cranor has proposed a general model to aid in understanding and designing out security problems that may arise from the interaction between humans and software systems: the Human-In-The-Loop (HITL) Framework [24]. The Human-In-The-Loop framework refines the C-HIP model for computer security communications. Both models describe a set of sequential stages with feedback loops that the receiver of a dialog might experience, with flow of information or processing from one stage to the next, until a change of behavior attributable to the communication takes place.

The framework describes five types of communications: *Warnings*, "communications that alert users to take immediate action to avoid a hazard"; *Notices* and *Status indicators*, which deliver information about relevant objects and their states; *Training communications*, aimed at teaching

users what to do to avoid threats; and *Policies*, which inform users of organizational or system requirements that are to be met during the operation of the system [24]. The HITL framework applies to all five, but in this thesis I focus on the first category only (*warnings*), as applied to computer security dialogs.

In Figure 2.5, there is a communication to be delivered to a human receiver in the form of a security dialog. Assuming that the communication has not been interfered with or distorted before reaching its human receiver, the dialog is processed in several steps. The receiver may or may not switch her attention towards the dialog. If she does, she must attend to the dialog long enough to comprehend its meaning. If the meaning is grasped, the receiver must acquire and retain the information from the dialog, and apply it to the current situation. The process ends when some behavior attributable to the dialog is observed. The whole sequence can be modulated or even completely overriden by the person's intentions, capabilities, or personal variables, which include her previous knowledge and past experience.

The framework includes an iterative process to identify and mitigate human threats in a system with four distinctive stages: task identification, task automation, failure identification, and failure mitigation. The framework also offers a set of questions that designers may use to find failures systematically in the interaction between users and a system, and to identify those areas that need improvement. A semi-linear set of stages, similar to those presented by the C-HIP model, allows a software designer to identify the part of the risk communication delivery that might fail.

2.1.5 Why we ignore computer security dialogs

Although not mentioned explicitly in the HITL framework, habituation to computer security dialogs is mentioned repeatedly as a problem in usable security research [11, 18, 24, 34, 47, 85, 87] [2]. Prior work has found (not without confounding variables) that web browser dialogs that resemble those that participants frequently need to dismiss are more likely to be ignored by participants in lab experiments than less-familiar designs [34, 85, 87]. Authors have also reported about habituation that begins to occur after just a few exposures to a new dialog [11, 18, 87].

Despite the previous discussion, the evidence to date about habituation has been mostly circumstantial. Studying habituation is extremely difficult, as it requires exposing participants repeatedly to a stimulus as part of the experiment. Among other problems, this makes it very difficult to conceal the purpose of the study and introduces learning effects that can be balanced but not canceled out. None of the previous studies were specifically designed to measure the impact of habituation, so they did not completely control for other factors that might have been responsible for users' behavior, and did not measure the impact of varying the number of repeat exposures. To the best of my knowledge, the only attempt to measure habituation was that of Kim and Wogalter: they found, in a lab experiment, that habituation occurred when participants were repeatedly shown the same computer dialog design. Exposure to a new design resulted in an increase in alertness; however, a return to a habituated design resulted in "recovery" of habituation [50].

In the next sections, I briefly describe habituation, and I offer two theories that may help explain why it occurs.

[2] Karlof et al. [47] use the term *click-whirr responses* instead of habituation; however, a careful reading of their paper does not suggest a meaning different than the one I provide for habituation in the previous section.

Habituation: how we become used to security dialogs

When non-compliant behavior does not cause harm over time, people may develop an automated response, *habituation*, that does not take into account changes in security dialog context or messaging [46]. Habituation decreases dialog effectiveness when people become less alert to the information presented in dialogs.

Habituation has been defined as "reduced attentional response to repeated exposure to a stimulus" [50]. Attention is a partially conscious, partially automatic process [3, 32, 100]. In the particular case of visual attention, the human cognitive system is continuously selecting the most salient object within the visual field, and directing the gaze to the selected object. Once there, the cognitive system extracts visual features from the object (size, shape, color, etc.), encodes those features, stores them in working memory, and searches the long term memory for similar encodings. If an encoding is found, then the object has been recognized, memory for the object has been formed, and its relative salience decreases compared to the rest of the objects with which it is competing for attention [3]. The process reinforces itself: the more times we recognize an object, the faster we recognize it in the future, and the more likely it is that we will miss some of the object's features that are lower in the hierarchy of salience. This is what we refer to as *reduced attentional response*.

Computer dialogs, just like their physical counterparts, have *iconic* and *informational* elements [32]. In a warning, iconic elements are size, color, icons, typography, geometry, etc. The informational elements are those that communicate a message to the receiver of the warning. The distinction is blurred, and some elements like the main warning word (e.g., 'Danger!') may incorporate both. While the iconic aspects in a computer security dialog are directly related to salience, the text within a given dialog is usually not. Habituation occurs when a person recognizes the iconic elements and prematurely stops processing the informational elements in the dialog. The problem is worsened by the fact that most systems have standardized the appearance of dialogs, or have at best a limited number of different *templates*, in the sense explained in Section 2.1.3. Visual variability between different messages is accordingly very limited, which may increase the likelihood of appearance of habituation, or reinforce already existing habituation.

Scripts-based understanding theory

Although all aforementioned authors agree that overexposure is the most likely cause for habituation, one alternative explanation can be found in *scripts-based understanding theory* [77]. A *script* is an ordered sequence of actions or events that people adopt when interacting with a class of objects [72]. Scripts are partially automatic, and are 'invoked' whenever the triggering conditions of a script are recognized. The more a script is rehearsed, the less conscious the process is, and the less attention is paid to each of the steps, which, in the case of interactions with warnings, includes attention to the warning itself [92]. Changing the appearance of warnings [18] or showing them less frequently has been shown to reduce habituation, because the warning does not match the stored script anymore [99]. Making a warning more prominent, for example larger or brighter, can help attract attention to the warning [100]. This theory suggests that making a computer dialog visually different can temporarily disrupt habituation; however, it remains a challenge to redesign users' interaction with security dialogs to prompt users to deviate from their already-acquired scripts.

The False Alarm Effect: how we perceive warnings as unimportant

Human beings are able to predict the consequences of threatening events in the future with varying degrees of accuracy which, paradoxically, gives us some calmness and comfort and at the same time produces anxiety for the uncertainty involved in the process. Threat is defined as "a stimulus conveying information about a future event with which a negative affect is associated" [17]. To be useful, any warning detection system must give its users the ability to protect themselves from the harm, and an appropriate time frame to do so. If one of these two conditions fail, the warning detection system is useless. Knowledge about the time frame without the ability to avoid it produces an "incubation of the threat," which results in fear and anxiety [17]. All warning systems must trade off accuracy for prediction time frame: the longer the time frame, the less accurate the prediction. This happens because the more sensitive a system is, the weaker the signals it detects and the longer the time frame it gives before the actual danger materializes. At the same time, the likelihood of an event inversely depends on the time frame prior to the event. Furthermore, how effective a warning system is depends on its perceived reliability, and this depends in turn on how many false alarms the system has produced. Every failed forecast (false positive) reduces the perceived reliability of the system. Usually, a correct forecast restores to a large extent the trust on the system. This phenomenon is known as the *False Alarm Effect* [17]. Every repeated false alarm decreases the user's reliance either on the warning system or the perception of the danger itself.

How false alarm effect and habituation reinforce each other

Many computer security dialogs are effective at communicating that there is a problem, but do not communicate clearly what the problem is or what the user should do to avoid it. For example, dialogs often include technical terms that an average user is not able to understand [66, 74]. In terms of the efficacy of the security dialog, this is equivalent to not providing a way to avoid the risk. In terms of the false alarm effect described above, such dialogs become threatening stimuli without a practical way to avoid them, thus producing incubation of the threat. If the dialog also turns out to be a false positive, we are in a situation where habituation and false alarm effect reinforce each other. As a result, users quickly learn to recognize and dismiss these computer security dialogs because, in their perception, they are not only threatening but also deceptive.

Consider an SSL security dialog, like the one shown in Figure 2.1. Many websites implement SSL certificates incorrectly [87], which normally trigger a security dialog without any real threat. Every time an SSL security dialog is shown without any real threat, the user is desensitized a bit to future SSL dialogs, and her reliance on the dialog (or the computer system as a whole) decreases. There exists evidence of this effect, both by using eye-trackers [6] and by asking users who have just dismissed security dialogs if they saw any security dialogs [85]. In a web browser phishing warnings study, authors found a significant negative correlation between recognizing a dialog and the willingness to read it, and a significant positive correlation between recognizing a dialog and ignoring it [34].

2.2 Related work

2.2.1 Evidence of attention failures

In the physical world, people pay sporadic attention to warnings, and are particularly likely to ignore those that do not map well onto a clear and understandable course of action [96]. Similarly, computer users systematically ignore security dialogs for a number of reasons. In this section I present a small subset of the numerous studies that provide evidence of security dialogs being ignored, and I explore very briefly at what point within the HITL framework security dialogs (or other security indicators) fail.

In their seminal work, Dhamija et al. analyzed an archive of 200 examples of phishing attacks and reported on three strategies used by phishers to fool computer users [27]. These strategies were *lack of knowledge*, *visual deception*, and *bounded attention*. The last category included two sub-categories: lack of attention to security indicators, and lack of attention to the absence of a security indicators. The authors discussed how security is to most users a secondary goal, and how this may lead to security indicators that are simply not noticed, not looked for in the right places, or not looked for at all [27].

Assuming that the communication channel is not tampered with (a problem that we will briefly describe in the next section), the reasons for ignoring warnings may be grouped into four categories:

1. *Failures in personal variables*: A user may not have the knowledge or the experience that is required to respond to a security dialog.

2. *Failures in intention*: A user may be unmotivated to respond to a security dialog; either because she believes that the dialog is irrelevant compared to her primary task, that it is not urgent to respond to the dialog, or that the message does not apply to her.

3. *Failures in communication delivery*: A security dialog fails to capture a user's attention, either because it is not salient enough compared to the rest of the stimuli in the environment, or because the user has been conditioned to ignore the warning.

4. *Failures in communication processing*: A user does not understand the message being communicated, or the options available in the dialog.

Failures of attention in usable security studies

Research conducted by Egelman et al. reviewed most stages of the C-HIP model to explain why a large number of participants were fooled by spear phishing messages sent to them in a lab study. Participants enrolled in a 'shopping study' were asked to buy two pre-defined products from Amazon and eBay with their own credit cards. After each purchase, participants were given a set of questions about shopping to increase credibility on the ruse; while doing so, researchers sent spear phishing emails to participants with URLs pointing to spoofed versions of eBay and Amazon. Emails asked participants to go to the corresponding website to confirm the order. Participants who clicked on the links saw the phishing warning corresponding to his/her assigned condition. If they proceeded, they were asked to enter their username and password. Researchers

found that active dialogs performed significantly better than passive ones. About half of participants answered correctly the question "what is phishing?"; the authors found a significant positive correlation between answering this question correctly, reading the dialog, and heeding the dialog [34]. Although this study was able to identify specific stages of the model at which participants failed, it did not directly address why they failed.

Wu et al. reported on two studies performed to determine whether users of three made-up web browser security toolbars (inspired by five existing security toolbars) paid attention to the security dialogs displayed to protect them against phishing attempts. In the first study participants were sent 20 emails; 5 of them were phishing messages. 67% of participants were fooled by 4 out of 5 phishing emails sent. The second study was a follow-up to test whether active indicators (that is, modal dialogs, or dialogs that prevent the user from taking any other action until she responds to the dialog) were effective at preventing users from entering their information. 7 out 10 subjects in the control group and 4 out of 10 in the experimental group were still fooled by the phishing messages. The study advises using active dialogs instead of passive ones to prevent phishing, and not to use pop-up dialogs too often since this might decrease their effectiveness over time [101]. Participants failed to look at the toolbars (i.e., failure in *attention switch*), and others dismissed the dialogs as unimportant because web content looked legitimate (i.e., failure in response-efficacy, part of the *intentions* component). Participants also dismissed or rationalized dialogs away (i.e., failure in *personal variables*, and possibly in *intentions*.)

Schechter et al. conducted a study where 67 participants were asked to connect to online banking accounts and to perform several tasks, some using their own credentials, some using provided credentials. The website was modified to take out some key security indicators. The purpose of the study was to investigate whether people proceeded to the site in the absence of these cues. All participants ignored the first indicator (http protocol instead of https), 96% ignored the second (site-authentication images), and 56% ignored the third (a warning page instead of a username-and-password-entering page). The authors also found that role-playing led participants to be less risk-averse [78]. Although it is not clear from the paper, it can be inferred that participants saw the indicators but dismissed them, which would correspond to a failure in attention maintenance.

Sunshine et al. performed a study to understand how users react to SSL dialogs. The study was comprised of two parts: an online survey with approximately 400 users, and a lab study with about 100 participants. In the online study participants were assigned to nine different conditions, resulting from three different security dialogs in three web browsers. Between 30 and 60% of the subjects reported that they would proceed to a site if they were shown those dialogs. In the second part two new dialogs were designed based on existing ones and then shown to participants. This second part showed that active dialogs (i.e., that interrupt the user's primary task) are more likely to stop user's intention to proceed to the risky sites than the passive ones. An interesting result was that about 30% of people reported that they had seen the invented dialogs before, which also may point to some degree of habituation [87]. The authors explicitly analyzed comprehension of presented dialogs, and concluded that in 2 out of 3 dialogs, there were significant correlations between comprehension and ignoring a dialog. Additionally, in some cases expertise made a difference in terms of ignoring the dialog (i.e., a failure in knowledge and experience, part of the *personal variables* component.)

Sotirakopoulos et al. [85] attempted to replicate the study by Sunshine et al. in order to mitigate several problems with their experimental design. The problems considered most important were

the population sampling method and the assignment of web browsers to participants: participants in the study by Sunshine et al. were students from Carnegie Mellon University only, and subjects were randomly assigned to web browsers, probably changing their usual behaviors and biasing them to be more cautious about presented dialogs. In addition, original authors modified the original Internet Explorer dialog they used, biasing subjects towards caution. Sotirakopoulos et al. used 5 conditions, resulting from the web browser used (Firefox 3.5, Internet Explorer 7) and the dialog used (native SSL dialog, invented SSL dialog), plus a fifth condition where they asked participants to use only Internet Explorer 7 regardless of their preferred web browser. They reported that they failed to confirm the results from Sunshine et al., and they raised serious doubts about the ecological validity of experiments performed in the lab.

Motieé et al. reported that 77% of all participants in a laboratory study did not understand the purpose of security dialogs and consented to a fake security prompt [64]; in the same study, 22% of participants with a high level of computer expertise did the same.

Finally, a number of authors have studied knowledge and comprehension as relevant factors in human response to security dialogs [34, 64, 87]. However, the very concept of understanding is overloaded and elusive: it may refer to understanding of why a dialog is being shown in a particular situation (*"I know why it popped up"*), understanding of the problem that motivates a dialog (*"I know what is the problem"*), understanding of how to solve the problem (*"I know how to solve/avoid the problem"*), understanding of the consequences of a problem (*"I know what will happen if I..."*), and so on.

Failures of attention evidenced by eye-tracker studies

I also reviewed a number of eye-tracker studies. These studies are the only way to obtain direct evidence about whether security dialogs are actually looked at; that is, whether dialogs succeeded in the *attention switch* stage of the HITL framework.

Whalen and Inkpen studied user attention to security indicators in web browsers while performing some common tasks, one of which being purchasing an item online with fake data. Participants were asked to treat the credit card and account data as if it were their own. Authors used an eye-tracker to determine what security indicators participants looked at. They concluded that the lock icon is noticed but not used, and that certificates are rarely used by users, if at all [94]. Most participants did not check whether pages were secure because they felt that data was not their own, so they were not motivated to take any security measures. In terms of the HITL framework, in this study there were failures of attention switch (participants did not look at some indicators, like the certificate icon), and attention maintenance (participants looked at the lock icon, but did not maintain their attention to it). Authors also mentioned that certificates were not understood by most participants, which counts as a failure in comprehension.

Sobey et al. tested improvements to extended validation SSL certificate indicators (EV indicators) in several web browsers, finding that while about half of their participants noticed a modification introduced to the web browser location bar, only a minority of users clicked on the location bar to obtain more information about websites being visited [83]. Similar to the previous study, participants failed to notice the EV indicators (i.e., failure in *attention switch*), and when they did, no action followed (i.e., failure in *attention maintenance*).

Bahr and Ford studied participants' reactions to nine pop-ups while engaged in security-unrelated tasks [6]. Authors asked participants to respond to unimportant prompts (e.g., '*you're*

the one-millionth visitor to our website, click here to claim your prize!'); they focused more on users' response time and ex-post affect reactions than in the security decision itself. The authors observed that participants were unmotivated (i.e., failure in *intentions*) to respond to dialogs, regardless of either the dialog presented or the level of engagement with the task. Since participants were forced to click on modal pop-ups with only one option ('OK'), lack of motivation is not surprising.

2.2.2 The Trusted Path problem

The messages that are communicated by security dialogs can be tampered with by malicious attackers. This problem is especially relevant in usable security research. One aspect of this problem is how we establish and maintain a *trusted path* between a principal that offers a service that a human wants or needs, and the human. In Chapter 5, I describe a study in which the prevalence of this problem was quantified; in this section I will describe the problem and some of the solutions that have been reported in literature.

Description

When a user interacts with a computing device, she may actually be communicating with a number of different principals, including the operating system (OS), installed applications, or websites. The security of many user experiences rests on the assumption that there is a *trusted path* from the user to the principal she is communicating with, and that the user can correctly identify (authenticate) this principal. It is assumed that there is a trusted path between the user and the OS for such purposes as authenticating with a shared secret (e.g. a password) and authorizing access to capabilities that may impact the security of the system.

The challenge in developing strategies to address the trusted path problem is that one cannot use lab studies, or even short-term field studies, to prove these strategies are resilient to attack. Users are habituated to security rituals over time through repeated conditioning. To study how participants respond to attacks, researchers must study participants who are already conditioned. In other words, proving that a trusted path ritual will resist real-world attacks requires deploying the ritual in the hands of users who will be conditioned to use it.

Given the extreme costs of establishing the security of a trusted path ritual, it is essential to learn as much as possible from what is not working today. Relying on subtle cues to establish that a window belongs to the OS does not seem to work.

The failures of individual trusted path mechanisms suggest that the problem is unlikely to be addressed adequately by any single mechanism. Rather, future work may focus on rituals that combine mechanisms. For example, consider a ritual in which users must first enter a shared attention sequence and then expect to see a visual shared secret. These two mechanisms may complement each other: the expectation of a visual shared secret may make it harder for an attacker to trick the user into entering a password (or performing a security-sensitive action) without first providing the secure attention sequence to make the visual shared secret appear. The secure attention sequence may make it harder to trick the user into believing the visual shared secret is unnecessary—the user need only enter the secure attention sequence to check if it can be made to appear.

Finally, even if a trusted path ritual can be created, the existence of the ritual alone will be insufficient to protect users. So long as users are regularly asked to provide credentials or perform security-critical actions via other paths, they will be habituated to do so when the next attack comes.

Related Attacks

The spoofable credential-entry windows we examine in Chapter 5 are one instance of a trusted path vulnerability. More familiar examples are phishing of website credentials, in which both emails and websites are spoofed, and scareware, in which attackers spoof infection alerts that appear to come from already-installed trusted software to trick users into installing malware posing as antivirus software.

Felten et al. provided the earliest demonstration of a web spoofing attack, describing it as allowing "an attacker to create a 'shadow copy' of the entire World Wide Web" to observe user behavior and capture user information [36]. The bulk of web spoofing attacks take the form of phishing and its variants, in which attackers spoof email, text messages, voice, and other communications channels to lure victims to spoofed websites. When users login to the spoofed website, their credentials are sent to the attacker. Phishing succeeds because many users cannot, or do not, authenticate websites. Many users instead rely on the content area of the page, assuming that the look and feel of a website are difficult to copy [27]. Such users fail to properly interpret indicators of website authenticity [29]. Increasingly sophisticated spoofing attacks have been implemented, including attacks on more modern web browsers [102], attacks that use a graphical element to cover up the SSL lock icon [54], and attacks that spoof the entire web browser window, including the certificate functionality [56].

Phishing attacks are similar to the attack in our work in that they use spoofing to steal credentials, motivating a user to login via a URL that leads to the spoofed site, and then convincing the user to trust the spoofed site. Scareware attacks are similar to our work in that they often spoof windows that appear to come from trusted client software. Scare tactics motivate users to install fake antivirus software by creating the illusion that the client is already infected with malware. Once installed, the fake antivirus software is used to trick the user into paying to keep the software 'up to date'. Stone-Gross et al. summarize the methods and economics behind such fake antivirus attacks [86].

One of the challenges in understanding the scope of the scareware problem is that fake antivirus software is installed not only through social engineering scare tactics, but also through vulnerabilities, including web browser vulnerabilities that allow attackers to perform 'drive-by downloads'. In the one-year period from July 2008 to June 2009, Symantec reports having received 43 million attempts to install rogue security software [88] (As of December 2013, Symantec has not released more up-to-date statistics.) Rajab et al. claim that fake antivirus attacks date back as far as 2003, and found that, from January 1 2009 to January 31 2010, such attacks were increasing as a percent of domains containing malware from an incidence rate of 3% to 15% [70]. The fraction of fake antivirus sites that use social engineering as an installation mechanisms also increased to 90% [70].

The best publicly-available statistics on the rate at which users are tricked by scareware come from Cova et al. They discovered servers hosting rogue antivirus campaigns that reported back to the attackers event counts such as the number of users who downloaded the scareware [22]. The

researchers discovered that 7.7% of users who received javascript that simulated an antivirus scan initiated a download of the scareware. The actual per-attack success rates were lower as downloads may be aborted before installation commences. Only 5% of machines that presented the fake scan reported back to the attacker's infrastructure that installation was successful (roughly two thirds of the 7.7% that downloaded the scareware.) [21].

Related defenses

Operating system designers have been aware of the need to defend against trusted-path vulnerabilities since at least as far back as the early 1970's when Saltzer and Schroeder presented the need for a 'secure' path in the context of a scenario in which a user grants permissions (capabilities): "one thing is crucial–that there be a secure path from Doe, who is authorizing the passing of the capability, to the program, which is carrying it out." [76]. More recently, Ka-Ping Yee described trusted path as requiring "an unspoofable and faithful communication channel between the user and any entity trusted to manipulate authorities on the user's behalf." Yee highlighted the secure attention sequence in Windows (ctrl-alt-delete) as an example solution to the trusted path problem for credential entry [104].

Many of the defenses that protect users from spoofing attacks today rely on detecting bogus emails and blacklisting software and websites; they do not address the underlying trusted path problem. Such defenses are necessary because preventing spoofing not only requires a technology to support a trusted path, but also a change in user behavior to avoid untrusted paths. Users must unlearn the habit of providing credentials into windows they they cannot authenticate. This will require time and a clear set of rules that users can apply to reliably differentiate the OS from other principals.

There are three major categories of solutions to establish trusted paths: dedicated IO, visualizations of shared secrets, and secure attention sequences.

Dedicated IO: One way to establish a trusted channel between the user and an OS is to dedicate specific hardware, or portions of hardware, to be used exclusively for that channel. For example, a device could dedicate a screen and separate keypad for use in authentication, as is sometimes done in payment systems. Others employ a separate input and output device that the user may already have. For example, Parno et al.'s Phoolproof Phishing scheme employs the user's mobile phone to externally confirm websites when entering a password [67] and IBM's Zone Trusted Information Channel provides an external trusted path for banking operations [79].

Enabling users to communicate securely with a single principal need not necessarily require both a dedicated input and output device. A dedicated output, such as an LED or dedicated screen region, may be sufficient to indicate when a trusted path is present. A dedicated input device may be sufficient to force a trusted path to be established, or may itself be used for the sole purpose of entering credentials.

Many systems attempt to establish trusted paths by dedicating pixels within a window to signal the presence of a trusted path. For example, the "chrome" region in web browsers is the portion of the browser window that is not controlled by the website being rendered, and has been used to host indicators that activate when a connection is secure or that display the domain name of a website. However, users can still be confused about whether a window is real or fake. For example, Jackson et al. demonstrated that users will trust spoofed chrome elements that appear in

a web browser window that is itself spoofed, rendered within the content region of a genuine web browser window [45]. This is known as a picture-in-picture attack.

Operating systems sometimes use visual cues to differentiate active windows (those 'in focus') from inactive ones, in part to defend against picture-in-picture attacks. For example, some systems render the frames of foreground windows to appear darker than background windows. Secure windows management systems EROS [80] and Nitpicker [37] dim all windows except the application currently in use and clearly label windows to help prevent users from accidentally entering information into the incorrect application. The results of the experiment we report about in Chapter 5 raise doubts as to whether dimming the screen is an effective way to establish a trusted path, and if an entire window can be spoofed, the labels inside can be as well.

Visualizations of shared secrets: While operating systems allow other principals to use the screen, they can usually ensure that they themselves can render data to the screen without it being intercepted by other applications. Thus, if the operating system and user share a secret, the OS can display this secret with reasonable confidence that other principals will not learn it. Shared secret schemes work much in the same way a dedicated output device does, but instead of lighting up a dedicated set of pixels to signal a trusted path, the OS renders a representation of the shared secret.

Tygar and Whitten propose "requir[ing] the consumer to personalize the appearance of the software at the time the trust relationship is formed" for this purpose [91]. Similarly, Adelsbach et al. suggest personalizing security indicators in the web browser interface [1]. Dhamija and Tygar's Dynamic Security Skins tool displays a user-selected photograph in windows requesting or providing security information, allowing users to verify that the window was produced by the web browser and not a website [26]. In Herzberg and Jbara's Trustbar, users assign names or logos for each website, and these are later shown to confirm that the users are again at the same website [42].

Other solutions use secrets that are not directly controlled by the user. Ye et al. present a colored border for pop-up windows to indicate when these windows are controlled by the web browser. The border format dynamically changes to match a browser-controlled metadata window [103].

The security of shared secret schemes rests on the assumptions that users will be able to recognize the shared secret, notice when the shared secret is absent, and realize there is no trusted path when the shared secret is absent. Shared secret schemes may be attacked by convincing users to disregard an invalid or missing secret. For example, Schechter et al. demonstrated an attack against the Passmark shared-secret scheme used for online banking in which users were told that their shared secret was temporarily unavailable due to system maintenance [78].

Secure attention sequences: Just as shared secrets leverage the operating systems' ultimate control over output devices, secure attention sequences leverage their ability to capture and prioritize input events. For example, on Windows the key combination of ctrl-alt-delete is captured by the operating system, and triggers the establishment of a trusted path to the OS, regardless of what applications are running. Since Windows NT, the OS has required that users unlock their computer with this sequence of keys, a *secure attention sequence*, before logging into the device. This sequence stops the execution of other processes, ensuring the existence of a trusted path for

the authentication process [65]. Unfortunately, Windows does not explicitly tell users not to enter their passwords without typing the secure attention sequence, and legitimate applications often ask users to do so.

A number of phishing prevention mechanisms have used secure attention sequences, including a 2005 proposal by Ross et al. [75]. Libonati et al. performed a field study to measure the efficacy of secure attention sequences in protecting web logins. No mechanism came close to being foolproof, even though participants in the study knew that they were being tested on their abilities to protect themselves from attacks, and given incentives to protect their passwords [57].

In summary, solving the trusted path problem is daunting. Providing dedicated input or output devices for authentication is impractical: it is costly, consumes device space, and would require a redesign of a myriad of devices. Trusted chrome has proven too easily spoofable. Establishing a trusted path via users' existing mobile devices for authentication simply passes the buck onto another general computing platform, which may also be vulnerable to spoofing attacks. Users forget to enter secure attention sequences when they do not appear to be necessary.

2.2.3 Attention to security dialogs

In Chapter 6, I present a novel technique to drive computer user attention to *salient fields* in security dialogs. Although to the best of my knowledge, our technique is indeed novel, there is some prior work that has attempted to solve similar problems, or the same problem in different ways.

In the previous section I reviewed proposed solutions to the trusted path problem. Most of these solutions can be considered as techniques aimed at disrupting existing user *scripts*, or at establishing new, trusted *scripts* (see Section 2.1.5.) For example, Dhamija and Tygar's dynamic security skins [26, 28] can be considered a technique to establish a script that the user can trust. Dynamic security skins do not *always* drive users' attention to important information in dialogs; instead, their application guarantees that whenever the conditions which were initially established to build the user script are not met (i.e., the user is not connecting to the server for which a skin was set), the appearance of the login box will look different, disrupting users' scripts, and bringing their attention to the problem at hand.

A number of studies have proposed techniques to raise computer users' attention when entering passwords in web browsers with weak or non-existent encryption. For example, Keukelaere et al. proposed that security dialogs should be adapted to the context in which they are shown, making the process more automatic and less demanding for the user [49]. This would help to decrease the amount of dialogs shown to the user, decreasing the level of habituation. Maurer et al. proposed to modify web browser forms to automatically detect boxes to enter critical information (e.g., passwords or credit card numbers). The authors show a contextual dialog that raises awareness of the information being entered [60].

Finally, a seminal work by Brustoloni and Villamarín-Salomón used two techniques to improve people's heeding of security dialogs: audited and polymorphic dialogs [18]. The first one was to warn users that their actions were being audited by a human observer who might impose penalties on them. In this condition, people made better security judgments related to their email, showing that it is possible to increase users' motivation (shown within the 'intentions' box in the HITL model, see Figure 2.5). However, auditing people's actions is resource-consuming and requires an organizational context that home users do not have. The second one was to randomly

change the ordering of options in dialogs on each presentation, forcing the user to actually read options presented to her, thus increasing her attention. Although this last technique can be applied to home users effectively to disrupt their existing scripts, it neither improves the quality of dialogs nor puts the user in a better position to make an informed choice. In terms of the human-in-the-loop framework, the first approach aims to increase participants' motivation, and the second aims to increase participants' attention. Despite these criticisms, this work remains a great inspiration for the techniques I present in Chapter 6.

Chapter 3

Bridging the gap in computer security dialogs: a mental model approach

3.1 Introduction

Computer security dialogs are intended to protect users and their computers. However, research suggests that these warnings may be largely ineffective because they are frequently ignored by users. This chapter describes a qualitative, role-playing lab study designed to gain insight into computer users' understanding of common security dialogs, the most frequent self-reported actions when a user faces a security dialog, and the ways in which advanced and novice computer users perceive and respond to security dialogs.

The mental models approach has been used in areas such as nuclear waste management, radon pollution, and sexual disease transmission [63]. A number of studies have applied mental models to computer security or privacy risk communication. Camp describes five generic mental models that might aid in delivering computer risk communication to lay users, and concludes that these models "can be used to improve risk communication," acknowledging that a user study should be performed to test these models [19]. Recently, Wash identified four mental models for the notion of 'hacker', and another four for the notion of 'virus', through open-ended interviews with a similar methodology to the one we use in our study [93]. Our study focuses on people's reactions and beliefs about computer dialogs.

In order to improve users' understanding of security dialogs, we first need to determine how users think about security dialogs. For this purpose we conducted 30 interviews, 10 with advanced users in security and privacy and 20 with novice users. We categorized and coded their answers and used these codes to create a mental model diagram that illustrates the gap in knowledge between novice and advanced users. In this chapter, we present our efforts to better understand this gap, and the ways in which we can apply this knowledge to the creation of more effective dialogs that may help novice users make safer choices.

This chapter is largely a reproduction of a paper co-authored with Lorrie Cranor, Julie Downs and Saranga Komanduri [13], including previously unreported material.

3.2 Study methodology

Our goal with this study was to obtain self-reported, advanced user strategies when dealing with security dialogs, in order to design more effective dialogs for novice users.

We collected examples of 29 security dialogs used in popular operating systems and application software and categorized them into four warning types: information deletion or loss, information disclosure, execution of malicious code, and trust in malicious third-parties. We picked one or two dialogs from each category: a disk space dialog, an email-encryption dialog, an address book disclosure dialog, an email attachment dialog, and an SSL certificate dialog. We created at least one 'scenario' per dialog in which we briefly described a situation that provided context for the dialog appearance. Four of the five dialogs are shown in Figure 4.1, and correspond to the same dialogs we would use later in the experiment described in Chapter 4.

We recruited 30 advanced and novice users to interview for our study. Mental model studies typically include 20 to 30 participants. This sample size is large enough to be likely to reveal, at least once, any belief held by ten percent or more of the population [63]. In this study we do not make inferential statements about quantitative differences between groups and thus there is no need for a formal power analysis.

Ten advanced users were recruited by direct email invitations sent to two mailing lists at our university. They were between 22 and 63 years old (mean=30.7, σ=11.8), and included two faculty members, five doctoral students in computer security, two research programmers, and one information security researcher. Advanced users were considered as such if they had either taken at least one graduate course in computer security or privacy, or had worked for at least a year on computer security or privacy projects. Most of our advanced participants had multiple years of security course work or experience. Past studies have found that even lower levels of expertise are sufficient for making significantly better security decisions, for example in the context of phishing [82]. All advanced users were given $10 and a chocolate bar as compensation for their time.

Novice users were recruited through messages posted on Craiglist[1] and fliers posted in bus stops around our university. Respondents were directed to an online screening survey. Those who worked in any field related to computer security or privacy, or who had taken at least one college-level course in computer security were excluded from participating. We selected 20 participants for our interviews. Their ages ranged from 18 to 57 years old (mean=32.6, σ=11.6), and their occupations were diverse: seven students, six employees or supervisors for different industries, three professional musicians, two self-employed, and two unemployed persons. All novice users received $20 as compensation for their time.

We conducted one-on-one, open-ended interviews with advanced and novice users. Appendix B contains the full script of the interview. At the beginning of each interview all participants were told that we were interested in knowing how they made use of computers and software, and that we were not looking for any particular answer. Interviews had seven segments: a brief general section about computer usage; five sections that asked about reactions to dialogs, and a final segment about demographics. In each 'dialog segment' we showed a dialog and read aloud a brief scenario that described a non-technically savvy friend asking the participant for help (see Appendix B). We then asked the following main questions:

1. Could you tell me what this message is?

[1] http://pittsburgh.craigslist.org/

2. What do you think will happen if your friend clicks on X? (we asked for all the options present in the dialog)

3. What do you think your friend should do?

We asked participants to explain their reasoning and any terms that they used, in order to bring out their thought processes. Any interesting observations were followed up on by the interviewer. The audio recordings of the interviews were transcribed verbatim.

Two investigators read the same five advanced users' transcripts independently, identified common ideas, and assigned a unique code to each of these ideas. The code lists were compared, and the differences resolved to create a single code list to be used with the remaining transcripts. A new set of transcripts was then read independently by each investigator and new common ideas were coded. This process was repeated until no new ideas emerged (i.e, no new codes were generated), which happened after having read 7 advanced transcripts and 10 novice transcripts. All transcripts were finally read again and coded with the agreed upon code list. Semantic relationships between ideas were also identified. The obtained mental model is presented as a diagram of these relationships in Figure 3.1.

The arrow to the left of Figure 3.1 shows the main 'stages' of the model. The diagram depicts three sets of tasks that a user performs after the dialog pops up: *observing*, *deciding*, and *acting*. In the first set, a user observes and considers any of several factors and events (variables) to try to understand what is being communicated to him through the dialog message. After these observations, the user attempts to diagnose the cause of the dialog. Then, one or more actions are taken to attempt to address the perceived problem. If the diagnosis was correct and the behavior was appropriate, then the problem is solved. Otherwise, the problem may persist or another problem may arise.

3.3 The mental model

Figure 3.1 shows a synthesis of the mental model. Advanced users' tasks are shown in yellow, and novice users' tasks are shown in blue. A detailed mental model for advanced users is presented in the Appendix A.

Arrows depict an observed and likely relationship between two tasks. For example, immediately after a dialog pops up, novice users often consider the look-and-feel of the dialog; if they find it suspicious, then they often judge that their computer might be infected or cracked, or if they are visiting a website, that the website might be a phishing attempt. In contrast, advanced users often consider recent actions that might have prompted the dialog, and will search for the dialog text on the Internet to determine its legitimacy.

The mental model represents a set of common lines of reasoning about computer dialogs. As such, it can be used to better understand the differences between how an advanced or novice user would think about a particular dialog. There are several ways in which this mental model can inform and improve the design of a dialog.

One way is to determine under what conditions a certain belief must be addressed before showing a security dialog. For example, unknown applications are hazardous due to at least two risks: the application might be a virus, and the application might access and misuse the user's

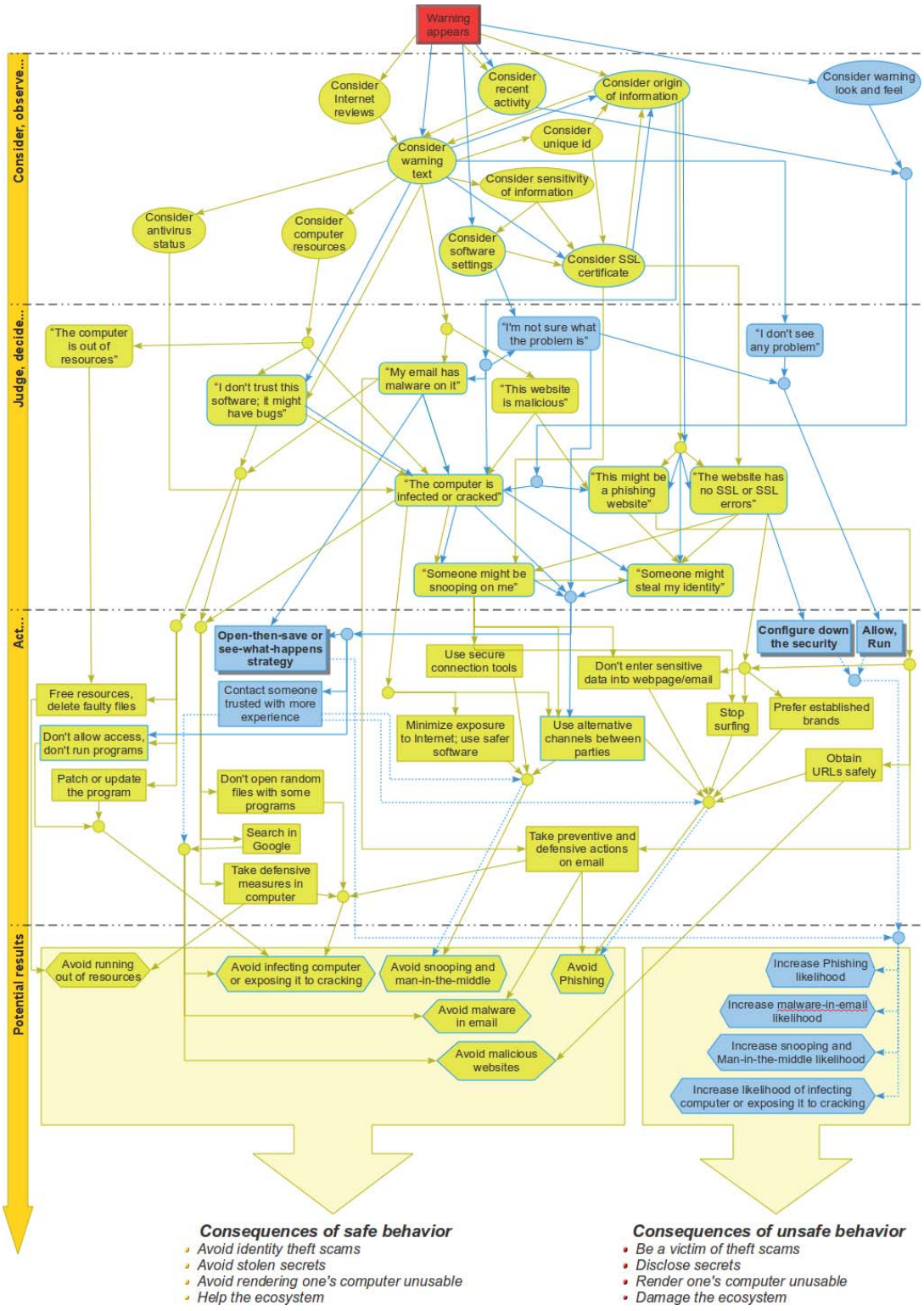


Figure 3.1: Our detailed mental model. Yellow items indicate advanced users' responses, and blue items represent novice users' responses. Yellow items with a blue outline were mentioned by both.

personally identifiable information. A smartly designed interaction would discard the first alternative by executing an antivirus program first. If the program reports that the application is free from known malware, then the dialog text can be tailored toward the second alternative, adding the information that the application has been checked and is free from known viruses[2]. Based on our interviews, this would be very helpful to novice users since they tend to relate all security dialogs to viruses. As figure 3.1 illustrates, novice users tend to consider a dialog and determine either that there isn't actually a problem, or that their computer has been infected. Though reliance on an antivirus program is not a perfect solution, it illustrates the potential of using a more holistic warning design approach to make dialogs more informative and less generic.

Another possible use of this mental model is to use it to prioritize between different messages to be delivered in a risky situation. For example, at the bottom of Figure 3.1, most of the consequences of unsafe behavior are caused by three actions of novice users: configuring down the global security level of the computer, allowing unknown programs to run, and a set of simple strategies represented by the labels 'open-then-save' and 'see-what-happens'. If several possible paths can be traced in the mental model, and the mental model shows that one of them might lead to an unsafe action, it is more important to discourage the user from taking this path than others.

Finally, insights from the model can inform the content of the dialog text itself. As the top sector of Figure 3.1 illustrates, novice users often do not consider the sensitivity of the information they enter into emails or websites, which makes them more likely to be victims of phishing or identity theft. This fact suggests that phishing dialogs should focus on the sensitivity of the information entered into an unknown website, rather than giving a vague signal that the current site might be a phishing site.

3.4 How advanced and novice users differ

We found consistent differences between advanced and novice users' behavior. One interesting difference is that both groups observe different cues, and as a result arrive at different conclusions about the risks they might be facing, which leads them to take different actions that will ultimately produce different outcomes.

We also observed more specific differences. For example, novice users assess the safety of an action after engaging in it, while advanced users judge how safe actions are a priori. It is probably unrealistic to change this behavior, but in many cases it is possible to offer a brief description of both the risks involved, and the consequences of each option offered in the dialog, thus cuing the novice user to consider this information in advance, as advanced users do. The information should be presented in a manner that makes it available to novice users but does not burden advanced users.

Also, novice users consider fewer factors and perform fewer tasks to ensure their safety, whereas advanced users perform notably more actions towards their safety. Advanced users look for vulnerabilities in public expert forums; they regularly patch and update their software; they use 'safe URLs' (e.g., from their personal bookmarks, recovering them by autocompletion from the browser surfing history, typing them directly in the location bar); they take proactive measures (e.g., maintaining antivirus programs, and installing security plugins in their browsers). Although it is unrealistic to ask a novice user to perform all these actions, it should be possible to gather and

[2]We tested this idea in one of the experiments described in Chapter 8.

display useful information to the user when needed. For example, all dialogs triggered by an email client might include a link to the online support forum maintained by the software vendor for that product, rather than to generic help text. This would empower advanced users to contribute their solutions to the problem that triggered the dialog, and these solutions would be an invaluable help to novice users. A dialog might also check automatically for available patches, or make use of heuristics to attempt to determine whether a typed email contains sensitive information. These and other strategies should be tested to evaluate their effectiveness.

Although one might believe that novice users simply 'hate' security dialogs, we did not find this to be the case: approximately half of our novice participants mentioned that they considered the presented dialogs as a 'good thing', regardless of their understanding, while the other half were neutral (none of them declared that presented dialogs were 'bad' or 'not useful').

3.5 Security misconceptions and problems

The responses of our novice participants revealed several security misconceptions. These misconceptions are indicative of inaccurate mental models, and illustrate how important it is to understand users when designing security solutions. For example, six novice users reported that their interactions with bank websites ought to be safe simply because banks have good security. In a scenario where a bank's website produces an SSL dialog – a scenario in which advanced participants strongly counseled against proceeding – two novice participants said:

> **Elizabeth**[3]: ... I would hit yes, yes ... I mean, assuming he trusts his bank. It's just, you know, the security certificate, you know, everything is valid about it, it's just you haven't elected to trust it yet, so I would feel better about hitting yes to that.

> **Michael**: Their site should automatically be secure because it's a bank. They're dealing with peoples' sensitive, private information like checking accounts, savings accounts, credit card information, Social Security information. That stuff is sensitive, so most banks should ideally have really complex security.

Two novice participants wanted to adjust the security settings on the computer to prevent this dialog from appearing, because they were so sure that a bank's website would be safe.

While advanced users agree that banks will have good security, they would not proceed to a bank's website if presented with an SSL certificate error. Advanced users are more likely to recognize the possibility that they are not truly at a bank website, whereas novice users rely on what might be fraudulent cues. As observed by Wu et al. [101], novice users often make security judgments based on look-and-feel, and our data supports this. When a novice participant was asked how he could tell that a dialog was authentic, he said:

> James: I guess the message looks authentic in terms of just the design, the icon used, and the font and the text and the gradient for the bar up top.

Eight of our novice participants cited the appearance of a dialog as a factor in deciding to trust it. This is in contrast to advanced users, who advised that appearance should only be used to decide not to trust a dialog, and never to confirm trust.

[3] All participants have been assigned pseudonyms of the appropriate gender.

Another misconception has to do with opening or saving files. When a security dialog presents a choice between opening or saving a file, advanced users feel that saving the file is safer because it can be scanned for malware before execution. By contrast, seven of our novice users felt that saving the file was more dangerous, since this permanently stored the file on the computer. They thought that opening the file only displayed a preview and was safe:

> **Melissa**: I would actually advise him to press 'Open' if he really wanted to see the chain e-mail because if you save a file that you're not sure would be safe or reliable, it's safer to open instead of save when you're dealing with something that you're not sure is reliable.

Four additional novice users perceived no difference between opening and saving files. These users felt that malware would activate either way. Joseph compared a suspicious e-mail attachment to a time-bomb:

> **Joseph**: Okay, a bomb or anything, I'm saying, okay, explode. Saving something, maybe I'm asking it to explode later.

A common problem found in computer dialogs is the use of technical jargon. Novice users often do not understand technical terms, which impedes their comprehension of dialogs. The security dialogs we used contained terms such as 'startup disk', 'encryption', 'virus', 'attachment', 'macro', and 'certificate'. Our participants had heard of, but not understood, these terms and struggled to make sense of them:

> **Stephanie**: I don't know whether if you send something that's unencrypted, does that mean that they can get into your whole computer and see everything. I don't, I don't know that. Can they see all your passwords and everything, everywhere you've been? You mean if something's unencrypted is it just the message or is it your whole computer that's kind of see through? I don't know.

SSL certificates turned out to be the most confusing concept in our study – sixteen novice users made incorrect statements about them. The SSL certificate dialog shown in this study indicated a website with a certificate that could not be verified. However, novice users associated this dialog with antivirus software, security updates or website certifications about being 'virus-free':

> **Michael**: Certificates are if you want to see or view how strong someone's computer security is from viruses. Basically, certificates say this is how you have programs like MacAfee and all these different, like, Norton Antivirus programs ... They're basically kind of a security guard against viruses.

> **John**: Oh, just, like, it has a valid name, a valid website, and it won't contain any harmful software, virus, or something else, and it could be trusted by any user or any other website.

> **Robert**: It is like, almost like a credential or like a plug-in that allows you to use software, and it means your security is up to date on your computer.

> **Melissa**: I guess it just proves how authentic a website is, whether (pause) I don't know how much the government plays, like, how much it monitors websites, but I'm expecting that it's a certificate from the government or company that says that website doesn't have any viruses, or that it's run by respectable people.

Neither the security dialogs that we showed to our participants, nor the brief scenarios that we presented along with each dialog contained the words 'virus' or 'security.' We believe novice participants used an availability heuristic [19, 63], assuming that viruses must be involved in any computer security context.

3.6 Conclusions

This study provides qualitative insights into how novice and advanced users make sense of dialogs. When presented with a security dialog, advanced and novice users observe different sets of cues, come to different diagnoses of the underlying risks, and consequently respond in very different ways.

This study suggests that in order to improve security dialog design, developers should consider all steps of dialog processing described in the HITL framework (see Section 2.1.4). Previous studies have considered factors like attention and motivation, which might improve users' awareness of different cues. However, our findings suggest that dialogs should also deal with wrong diagnoses by indicating, for example, when a specific condition was not produced by a certain problem (for example, novice users tend to overdiagnose virus problems).

There is a trade-off between the amount of information included in a dialog and the added likelihood that this new information might help users make the appropriate decision [97, 98]. We observed that participants in our study often did not read dialogs thoroughly: giving them more text to read may only worsen the problem[4].

To ensure that dialogs are presented only when necessary, and then with only the necessary information, we should insist on the application of the fundamental security dialog design principle: only present a security dialog when designing out or guarding against the risk is infeasible. In addition, security dialogs should be presented only in situations where the best course of action depends on details of the situation that are known to the user.

We have used mental models to highlight more effective ways to convey security information to the average user in response to immediate problems. However, it is also possible to make a more proactive use of these models to determine, for example, how to better employ users' time in security education or training. Mental models have myriads of different and unexploited applications in the study of the usability of security; one unexplored possibility is their use as Bayesian belief networks to automatically estimate the probabilities of the different possible risks. This may help developers in implementing heuristics to determine when a dialog is likely to be helpful.

A mental model approach is a valuable first step in assessing the usability of security dialogs. It starts with no assumptions about users' views and beliefs, and produces a model that directly reflects the knowledge and thought process of end users. Any aspects of the model can subsequently be tested by researchers using a more traditional hypothesis-driven experimental protocol.

[4] We tested this idea in the text-length experiment, reported in Chapter 8.

By guiding security and privacy researchers to areas where users require the most assistance, we see this research as a practical way toward protecting end-users and the systems on which they work.

Chapter 4

Improving Computer Security Dialogs

4.1 Introduction

This chapter reports on a follow-up study to the one presented in the previous chapter. In this study we explore the links between security dialog design and user understanding of, motivation to respond to, and actual response to computer security dialogs. We measured these variables through a 733-participant online study that tested a set of four existing computer security dialogs and two redesigned versions of each across low- and high-risk conditions. In some cases, our redesigned dialogs significantly increased participants' self-reported understanding and motivation to take the safest action; however, we were not able to show that participants' responses were differentiated between low and high risk conditions. We also observed that motivation seemed to be a more important predictor of taking the safest action than understanding. However, other factors that may contribute to this behavior warrant further investigation.

4.2 Methodology

We performed an online survey (N=733) to test the effects of dialog design on user understanding, motivation, and safe response. Our study used a 3 x 2 design, with three dialog design conditions (E: existing set of dialogs, G: redesigned based on design guidelines, and M: redesigned based on our previous work on mental models) and two scenario-based context conditions (S_1: low security-priming and S_2: high security-priming) for a total of six conditions.

We tested five existing security dialogs from commercially available software, but report on only four (one of them was not a security dialog). The four dialogs, referred to as the Existing set (E, see Figure 4.1), alerted users about problems encrypting an email (W_1), a program trying to access the user's address book (W_2), an email attachment (W_3), and an unknown SSL certificate (W_4).

We created a second set of security dialogs, referred to as the Guideline-based set (G, see Figure 4.3). Each of the dialogs in the E set were redesigned by three HCI Master's students who each had at least one year of HCI coursework as well as previous design experience. We asked

This chapter is largely a reproduction of a paper co-authored with Lorrie Cranor, Julie Downs, Saranga Komanduri and Manya Sleeper [16].

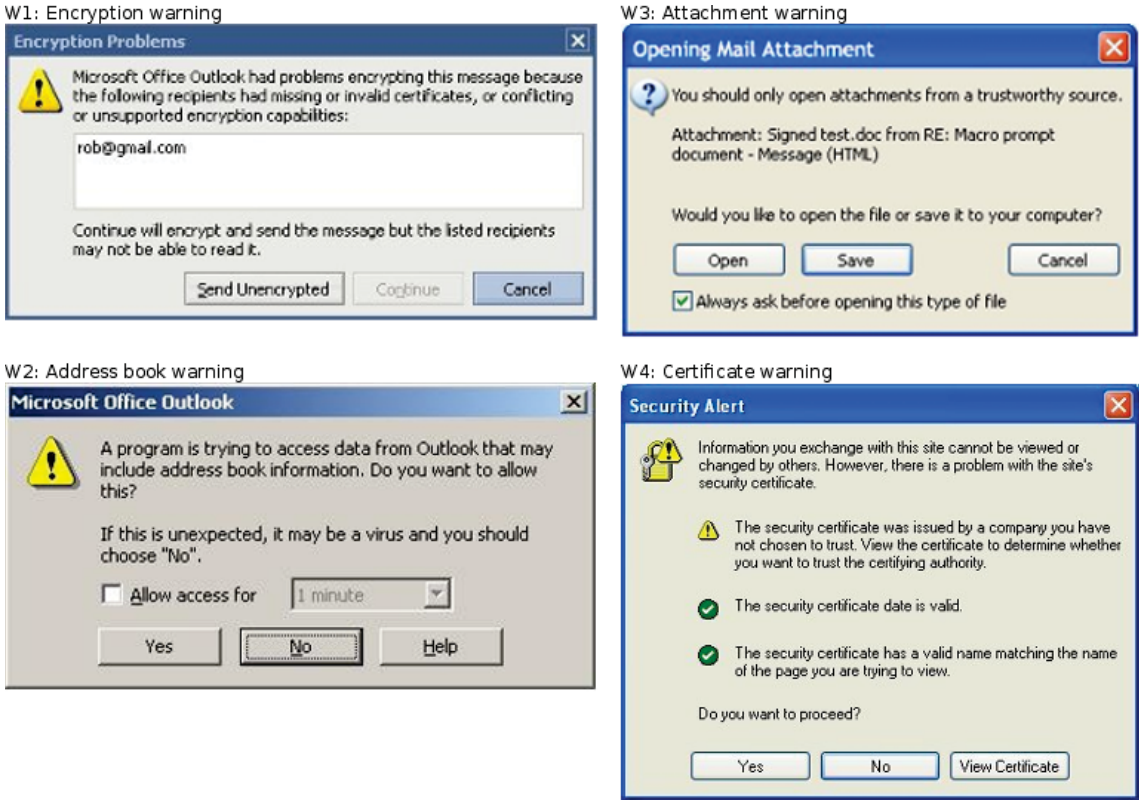


Figure 4.1: Existing (E) set of dialogs.

the students to redesign the existing dialogs by following the design guidelines we compiled from the literature [5, 9, 24, 33, 34, 62, 66, 74, 97]. A brief summary of these guidelines is shown in Table 4.1. We did not provide the designers with any other information about our study.

Similarly, we created a third set of dialogs, referred to as the Mental-model-based set (M, see Figure 4.2). To create this set we redesigned each dialog in the E set based on the work on mental models presented in the previous chapter. We tried to design this set of dialogs to include information that experts tend to seek out when responding to a dialog, such as the results of analyses by antivirus software. We also applied many of the guidelines used by the HCI students to create set G.

4.2.1 Contextual scenarios

Users view security dialogs within a specific contextual situation, and make a decision based on that situation. To imitate this context in our online survey, we wrote a low security-priming scenario (S_1) and a high security-priming scenario (S_2) for each dialog. Each user who saw a particular dialog was presented with a scenario along with that dialog. S_1 included low security-priming scenarios with activities that most people would not normally associate with a security threat; whereas, S_2 included activities that involved sensitive or confidential information, or had characteristics of common security attacks. As security dialogs must consistently be useful in both low- and high-threat contexts we chose to include both low and high security-priming categories

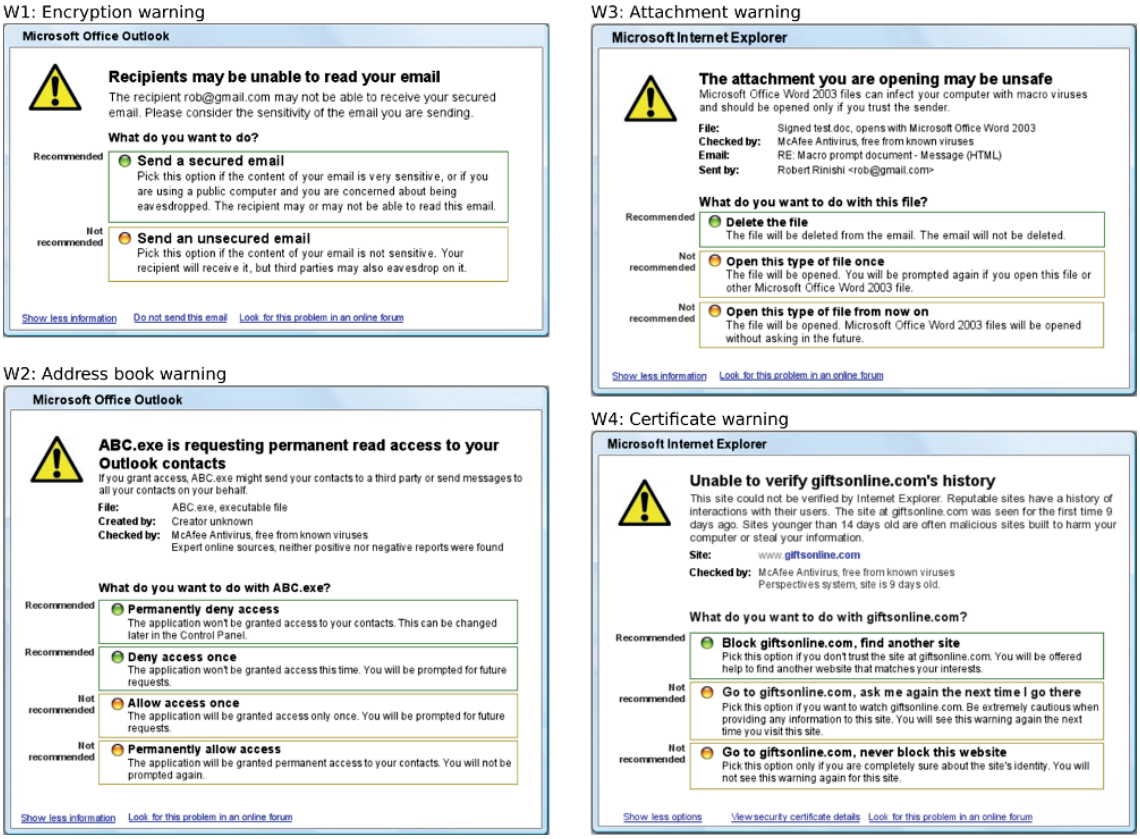


Figure 4.2: Mental-model-based (M) set of dialogs. This set was designed from the Existing set in Figure 4.1.

to ensure that our results were consistent across scenarios that presented different threat levels. Table 4.2 contains all scenarios. We incorporated feedback from security experts when creating the scenarios and strove to ensure that scenarios were of similar readability and length.

4.2.2 High and low risk conditions

Each dialog, in combination with each scenario, presented the user with either a high or low level of risk. Throughout this chapter, we refer to the level of risk that the participant faced when presented with a specific dialog and contextual scenario combination as either low-risk or high-risk. Based on our definition of safe response (see Section 2.1.3), when dialogs are successful, participants in low-risk conditions should choose not to take the safe response because the safe response requires them to sacrifice functionality. However, participants in high-risk conditions should choose the safe response because they should prioritize safety over functionality in risky situations.

All conditions are described in Table 4.2. We had two low-risk conditions: the encryption (W_1) and address book (W_2) dialogs with S_1 scenarios. In both cases the risk is minimal and taking the least risky action would prevent the user from completing her primary task. We had six

Figure 4.3: Guidelines-based (G) set of dialogs. This set was designed from the Existing set in Figure 4.1.

#	Guideline	Examples
1	Follow a visually consistent layout	Use one icon; do not use a close button; use command links for options; use a primary text to explain the risk; describe the consequences of each option below each button.
2	Comprehensively describe the risk	Describe the risk; describe consequences of not complying; provide instructions on how to avoid the risk.
3	Be concise, accurate and encouraging	Be brief; avoid technical jargon; provide specific names, locations and values for the objects involved in the risk; do not use strong terms (e.g., abort, kill, fatal)
4	Offer meaningful options	Provide enough information to allow the user to make a decision; option labels should be answers to explicit question asked to the user; if only one option is available, do not show the warning; the safest option should be the default.
5	Present relevant contextual and auditing information	If the warning was triggered by a known application, describe the application; identify agents involved in the communication by name; if user's information is about to be exposed to risk, describe what information and how it will be exposed.

Table 4.1: Guidelines used to redesign security dialogs.

high-risk conditions: all four dialogs with S_2 scenarios, and the attachment and certificate dialog

Dialog	Scenario	Text	Risk condition
W_1: Encryption dialog	S_1: Low security-priming	Imagine that you are sending a birthday greeting to your friend Rob by email. You click on the 'Send' button and the warning below appears on your screen.	Low risk
	S_2: High security-priming	Imagine that you are sending important financial information to your boss by email. Your boss warned you that it is important to keep this information confidential. You click on the 'Send' button and the warning below appears on your screen.	High risk
W_2: Address book dialog	S_1: Low security-priming	Imagine that you are trying to connect your PDA or smartphone to your computer to synchronize your email. You plug the device into your computer, and the warning below appears on your screen.	Low risk
	S_2: High security-priming	Imagine that you are reading your email. You open a message from your friend Rob, and the message invites you to try out a new social network your friend is using. You click on the invitation, and the warning below appears on your screen.	High risk
W_3: Attachment dialog	S_1: Low security-priming	Imagine you are reading your email. You open an email from a friend, who says that he is sending you a book he thinks you would find interesting. You double-click on the attachment, and the warning below appears on your screen.	High risk
	S_2: High security-priming	Imagine you are reading your email. You open an email that seems to be from one of your friends, but the email does not contain any text, only a document attached. You double-click on the attachment and the warning below appears on your screen.	High risk
W_4: Certificate dialog	S_1: Low security-priming	Imagine that you want to buy a gift for a very good friend, but you don't have time to go to a store. You look for a site on the Web, and after searching for a few minutes you find a website that seems to be OK. You click on the link to the website, and the warning below appears on your screen.	High risk
	S_2: High security-priming	Imagine that you need to pay a bill, and you are out of checks. A friend suggests you try paying the bill from your bank's website. You have seen your bank statements online before, but you don't know how to pay bills online. You remember you recently received an email from your bank. You open the email, click on a link to enter the bank's website, and the following warning appears.	High risk

Table 4.2: Low (S_1) and high (S_2) security-priming scenarios created for the study.

with S_1 scenarios[1]. In these cases, the level of risk warranted taking the safe response.

A well-designed security dialog should allow participants to differentiate between low- and high-risk conditions. It should create a higher rate of motivation and safe response for high-risk conditions than for low-risk conditions. If the dialogs in our study were well designed we would expect to see dialogs with the same level of risk in S_1 and S_2 (attachment and certificate dialogs) to have similar rates of motivation and safe response. We would also expect to see dialogs with low risk in S_1 and high risk in S_2 (encryption and address book dialogs) to have higher levels of safe response and motivation in S_2.

[1]The content of the attachment and certificate dialogs was suspicious enough to suggest a high-risk situation, even in S_1.

	W_1		W_2		W_3		W_4	
Scenario	S_1	S_2	S_1	S_2	S_1	S_2	S_1	S_2
E	145	124	114	145	125	114	106	125
G	119	106	124	119	145	124	114	145
M	125	114	106	125	119	106	124	119

Table 4.3: Number of participants per condition.

4.2.3 Survey design and participant recruiting

Our survey consisted of 69 questions divided into seven sections, starting and ending with demographic questions. Each of the remaining five sections included a randomly selected image of a dialog, a randomly selected scenario (S_1 or S_2), and a set of questions about each dialog.

We recruited participants using Amazon's Mechanical Turk service between July and August 2010, paying $0.50 to each participant who completed the study. We required participants to be computer users, over 18 years old, English-speaking and residents of the United States. Participants took an average of 10 min 47 sec to answer the survey (σ = 7 min 9 sec). We discarded 3 responses that took less than 10 seconds. We were left with 733 respondents, about 62% of whom were females and four-fifths of whom were Caucasian. The number of participants in each condition is summarized in Table 4.3. Participants ranged in age from 18 to 75, with a mean age of 32.9 (σ = 11.58). We also collected information about usage of operating systems, browsers, and email clients to test any correlation with our dependent variables. As described later, we found no consistent relationship between demographics and dependent variables.

We also asked two questions to probe participants' level of technical expertise: whether they had ever taken or taught a course on computer security, and whether they knew any computer languages. If they answered the latter affirmatively, we asked which languages they knew. Participants who answered only HTML were not considered as having programming expertise. We found no significant correlation between affirmative answers and any studied variables, so we excluded these questions from our analyses.

4.2.4 Hypotheses

To develop our hypotheses, we defined three dependent variables: understanding, motivation and safe response. These variables are described in Table 4.4. We also defined low- and high-risk conditions consisting of combinations of dialogs and scenarios, as indicated in the rightmost column of Table 4.2.

We hypothesized that understanding would be higher for all conditions in the redesigned dialogs than in the existing set. For motivation and safe response we hypothesized that they would be significantly higher in the redesigned dialogs for participants in the high-risk conditions but would not be significantly higher for participants in the low-risk condition. We also hypothesized that understanding and motivation would be found to drive safe response. Our hypotheses are enumerated below:

H1 : For all dialogs and scenarios, understanding will be significantly higher in the guidelines-based (G) and mental-model-based (M) sets than in the existing set (E).

Dependent variable	Question	Types of answers	Explanation
Under-standing	*What do you think is/are the problem(s)?*	11 common problems plus an 'Other' open text field	If participants answered at least one of the 'correct' answers and none of the 'incorrect' answers from a list based on our knowledge of usable security and on previous interviews with security experts [13], understanding was measured as 1, otherwise as 0.
Motivation	*The problem described by this warning is very important.*	5-point Likert question, from 'Strongly disagree' to 'Strongly agree.'	If participants answered as 'Agree' or 'Strongly agree', motivation was measured as 1, otherwise as 0.
Safe response	*What would you do in this situation?*	As many clickable options as the warning offered, plus 'Ignore this warning' and 'Take another action.'	If participants answered at least one safe action and none of the unsafe actions in a list based on experts' answers, safe response was measured as 1, otherwise as 0.

Table 4.4: Questions asked to participants per dialog, and the corresponding measured variable.

H2 : For all low-risk scenarios, motivation and safe response will not be significantly higher in the redesigned sets (G and M) than in the existing set (E).

H3 : For all high-risk scenarios, motivation and safe response will be significantly higher in the redesigned sets (G and M) than in the existing set (E).

H4 : Understanding and motivation will be significant predictors of safe response across all dialog sets and scenarios, controlling for demographic factors.

4.3 Analysis

Based on an analysis of the four dialogs we found that understanding and motivation were strongly correlated with safe response. However, we were not able to conclude that users could differentiate between low-risk and high-risk conditions, and we did not see a significant increase in motivation and safe response for W_1 and W_2 in either the high- or low-risk conditions. However, we did find improvements in motivation and safe response for W_3 and W_4, the two dialogs that were only presented in high-risk conditions.

We analyzed our results separately for each dialog using logistic regressions. We used a significance level of $\alpha = .05$ for all analyses.

4.3.1 Understanding

In general, our redesigned sets of dialogs (G and M) failed to increase understanding over existing dialogs. We observed significant increases in understanding in only 3 out of 16 conditions, and in

		E	G	M	E vs. G				E vs. M			
					c	*s.e.*	*z*	*p*	*c*	*s.e.*	*z*	*p*
W_1	S_1	41%	57%	66%	0.668	0.260	2.569	**.010**	1.025	0.260	3.945	**<.001**
	S_2	37%	48%	43%	0.423	0.274	1.542	.123	0.238	0.273	0.871	.384
W_2	S_1	68%	52%	34%	-0.696	0.278	-2.508	**.012**	-1.428	0.294	-4.859	**<.001**
	S_2	84%	91%	88%	0.648	0.408	1.590	.112	0.291	0.364	0.799	.424
W_3	S_1	62%	62%	65%	-0.027	0.258	-0.105	.916	0.125	0.273	-0.458	.647
	S_2	78%	81%	83%	0.178	0.328	0.541	.588	0.380	0.352	1.079	.280
W_4	S_1	74%	85%	79%	0.703	0.352	2.000	**.046**	0.279	0.318	0.878	.380
	S_2	73%	76%	83%	0.157	0.288	0.547	.585	0.596	0.324	1.840	.066

Table 4.5: Results of two logistic regressions comparing understanding levels between dialog sets. The two leftmost columns show the dialog (W_1 through W_4) and the scenario (S_1 and S_2) shown to the user. The next three columns show the proportion of participants who understood the problem that triggered the corresponding dialog, per dialog set (E stands for Existing, G for Guidelines-based, and M for mental-model-based). The next four columns show the parameters of the regression comparing the Existing and the Guidelines-based sets, and the rightmost 4 columns show the parameters of the comparison between the Existing and the Mental-model based sets. **c** is coefficient, **s.e.** is standard error, **z** is z-value, and **p** is p-value.

two cases related to W_2 we observed significant decreases in understanding. Figure 4.4 shows our results for understanding. Statistical data are given in Table 4.5.

We expected to see increased levels of understanding for the G and M sets versus the E set (H1). While this occurred in a few conditions, understanding did not increase in the majority of cases (see Table 4.8). Because understanding increased in more conditions in which participants were shown S_1 than S_2, we tested the possibility that participants spent less time on the scenarios by comparing the mean time that participants took to answer each dialog section in the survey. However, we found no significant differences between times for the two sets of scenarios.

In the S_1 scenario for the address book dialog (W_2), the understanding rate was significantly lower for the G and M sets than in the E set. To help explain this lower level of understanding, we looked at the specific problems that users thought the dialog presented. We found that a higher percentage of respondents believed that the dialog was related to a website in the G and M sets than in the E set, which was a 'wrong' answer. The misunderstanding was potentially due to a reference to **ABC.exe** (the program accessing the computer) that only appeared in the redesigned dialogs. We speculate that respondents may have mistaken **ABC.exe** for a website. We mandated in our guidelines that a program prompting a dialog should be identified to users, to help them better decide how to respond, but the implementation of this recommendation could have resulted in confusion.

The redesigned dialogs (G and M) were also less likely to prompt two 'right' answers than the existing (E) dialogs. For the G and M versions of the address book dialog in the S_1 scenario, participants were less likely to respond that they did not trust the software being run or that there was no problem than when shown the E version of the dialog. Participants may not have considered **ABC.exe** to be software, or perhaps they considered the redesigned dialogs more threatening than the existing dialog. Additional testing would be necessary to determine which aspects of the dialogs lead to misunderstanding.

These results provide very limited, if any, support for H1. It should be noted, however, that many dialog-scenario combinations had a high initial level of understanding, from which it may

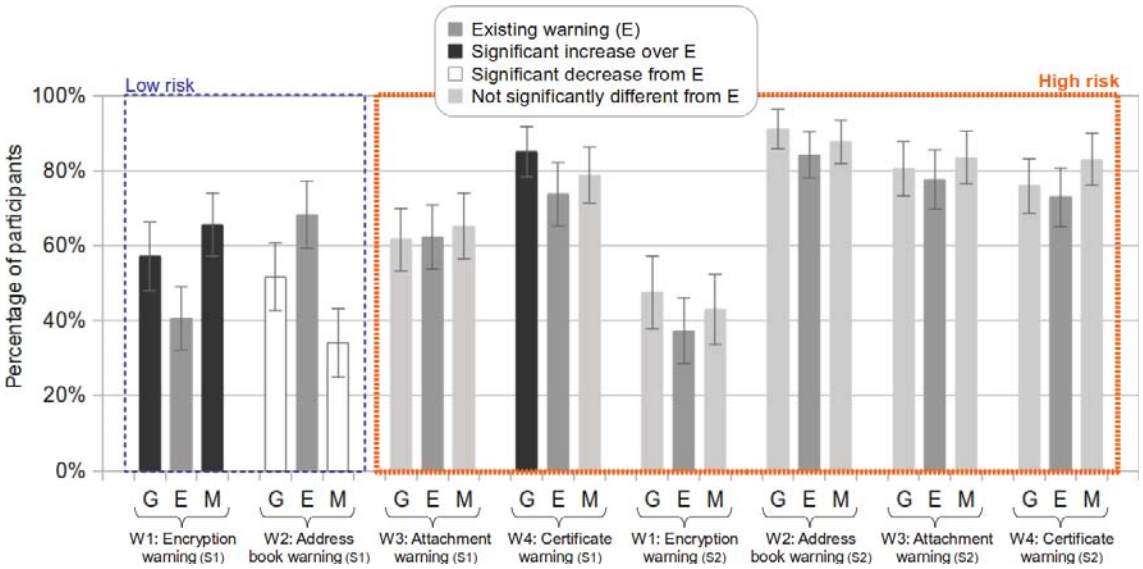


Figure 4.4: Percentage of participants who showed understanding of the problem that triggered the studied dialogs, in the low- and high-risk conditions. G, E, and M correspond to the different sets of dialogs. The top bars represent confidence intervals at the 95% level.

be difficult to introduce improvements.

4.3.2 Motivation

Our redesigned dialog sets (G and M) had some success at increasing levels of motivation in the high-risk condition for W_3 and W_4, but did not show evidence of allowing participants to differentiate between low- and high-risk conditions. Figure 4.5 shows our results for motivation. Statistical data are given in Table 4.6.

If the redesigned dialogs allowed participants to differentiate between high- and low-risk contexts and respond appropriately, there would be no change in motivation levels between G/M and E in the low-risk condition, but there would be an increase in motivation levels for the redesigned dialogs in the high-risk condition. We were not able to conclude that the redesigned dialogs allowed users to differentiate between low- and high-risk contexts. For the encryption dialog and address book dialog (W_1 and W_2), which were shown in both high- and low-risk contexts, there was no significant improvement in motivation in the majority of cases in either context.

In the low-risk context, we expected motivation not to be significantly higher for the redesigned dialogs (G and M) than the existing dialogs (E). This held for three out of four cases. However, for these results to be meaningful, we needed to see a corresponding increase in motivation for these same dialogs (W_1 and W_2) in a high-risk context, proving that participants could differentiate between the levels of risk with the redesigned dialog set and respond appropriately. However, we found that in all four high-risk cases for W_1 and W_2 there was no significant difference between the E set and each of the G and M sets for motivation. This indicates that the lack of improvement in the low risk case may have represented a lack of improvement overall, rather than participants' abilities to differentiate between risk levels. These results provide only partial

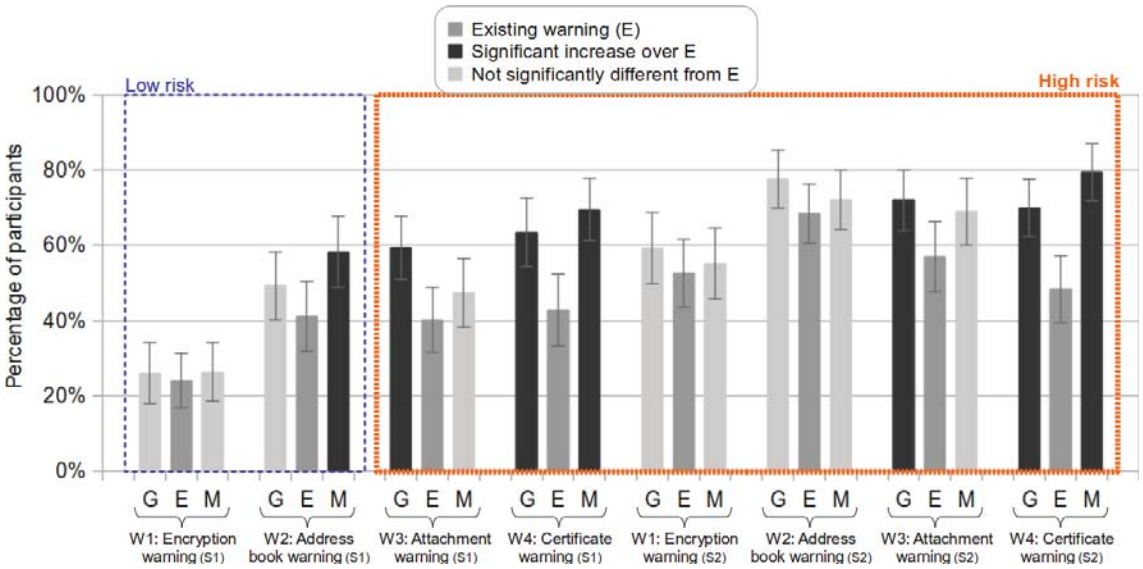


Figure 4.5: Proportion of participants who agree or strongly agree that the problem described by the dialog is very important (i.e., were 'motivated'), in the low- and high-risk conditions. G, E, and M correspond to the different sets of dialogs. The top bars represent confidence intervals at the 95% level.

support for H2.

Although there was no evidence that the redesigned dialogs allowed participants to differentiate between low- and high-risk contexts, we did find some evidence that the redesigns improved motivation in the high-risk context (H3). For the attachment and certificate dialogs (W_3 and W_4), which were only shown in high-risk contexts, we found that the redesigned dialogs significantly increased motivation in all but one case. As previously described, we expected to see similar results for W_1 and W_2 in the high-risk context, but did not see any significant differences between G/M and E for W_1 and W_2.

4.3.3 Safe response

We found that the redesigned dialogs were successful at increasing safe response in the majority of the high risk conditions. However, similarly to what happened with motivation, we were not able to conclude that the redesigned dialogs allowed participants to differentiate between high- and low-risk conditions and respond appropriately. Figure 4.6 shows our results for safe response. Statistical data are given in Table 4.7.

As described previously, 'safe response' measures the proportion of participants who pick the option that presents the least risk. We expected participants' rates of safe response to significantly increase for the high-risk conditions for our redesigned dialogs and to remain the same for the low-risk conditions. In the low-risk conditions the redesigned dialogs should not push participants to pick a safe response that would prevent them from completing the desired task. For the two dialogs that we presented in both the high- and low-risk conditions, W_1 and W_2, we found that, as expected, in three out of four cases, the level of safe response was not higher for the G and M sets than for the E set. However, for these two dialogs we also found that, in three out of four

		E	G	M	E vs. G				E vs. M			
					c	*s.e.*	*z*	*p*	*c*	*s.e.*	*z*	*p*
W_1	S_1	24%	26%	26%	0.098	0.296	0.330	.741	0.115	0.289	0.399	.690
	S_2	53%	59%	55%	0.271	0.272	0.996	.319	0.105	0.268	0.390	.696
W_2	S_1	41%	49%	58%	0.325	0.296	1.207	.227	0.692	0.280	2.470	**.014**
	S_2	68%	78%	72%	0.474	0.294	1.613	.107	0.178	0.275	0.647	.518
W_3	S_1	40%	59%	47%	0.779	0.256	3.049	**.002**	0.291	0.264	1.102	.270
	S_2	57%	72%	69%	0.664	0.283	2.344	**.019**	0.515	0.289	1.782	.075
W_4	S_1	43%	64%	69%	0.849	0.283	3.000	**.003**	1.117	0.282	3.956	**<.001**
	S_2	48%	70%	79%	0.909	0.262	3.472	**.001**	1.419	0.296	4.795	**<.001**

Table 4.6: Results of two logistic regressions comparing motivation levels between dialog sets. The two leftmost columns show the dialog (W_1 through W_4) and the scenario (S_1 and S_2) shown to the user. The next three columns show the proportion of participants who claimed that solving the problem that triggered the dialog was important or very important, per dialog set (E stands for Existing, G for Guidelines-based, and M for mental-model-based). The next four columns show the parameters of the regression comparing the Existing and the Guidelines-based sets, and the rightmost 4 columns show the parameters of the comparison between the Existing and the Mental-model based sets. **c** is coefficient, **s.e.** is standard error, **z** is z-value, and **p** is p-value.

cases, the level of safe response did not increase in the high-risk condition for G and M compared to E, indicating that the lack of improvement in the low-risk condition may have been due to an overall lack of improvement rather than participants' ability to differentiate between risk levels. So, although we found some evidence for H3, our overall results for safe response for dialogs W_1 and W_2 were inconclusive.

We did, however, find a significant increase in safe response levels for the redesigned dialogs (G and M) over the existing set (E) for the two dialogs that were presented in only the high-risk condition, W_3 and W_4. For these dialogs, rates of safe response significantly increased in seven out of eight cases. This result provides some support for H3. We performed a qualitative analysis of participants' open comments at the end of each dialog to test the possibility that these higher levels of safe response were due to the novelty of redesigned dialogs. We found no evidence of such behavior.

4.3.4 Correlation between variables

We hypothesized that understanding and motivation would be predictors of safe response (H4). We found significant correlation between safe response and understanding, motivation, and other variables (see Table 4.8), supporting H4. The higher logistic regression coefficients show that safe response is strongly tied to motivation and also linked, although slightly less strongly, to understanding. These results do not prove that understanding and motivation drive safe response, but they do provide some indication that the variables are strongly related.

Motivation and understanding were significantly correlated with each other for all dialogs. Motivation was significantly correlated with safe response for all four dialogs within each dialog set. Understanding was significantly correlated with safe response for all except the encryption dialog (W_1). Based on the regression coefficients, motivation was more strongly correlated with safe response for all of the dialogs in which both factors were significant, except for the address book dialog (W_2).

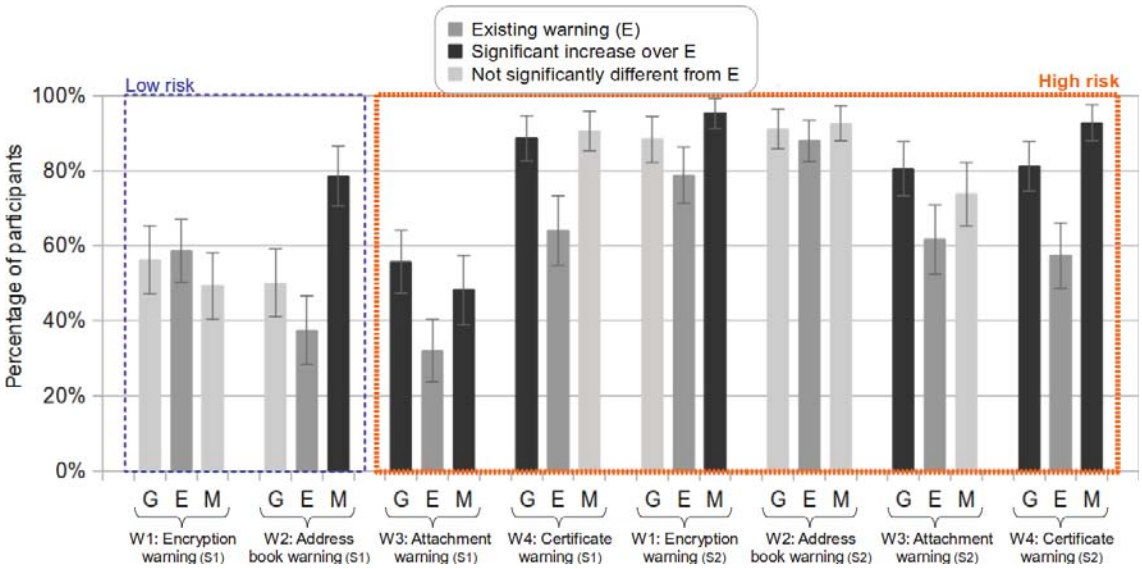


Figure 4.6: Proportion of participants who picked the safest option, in the low- and high-risk conditions. G, E, and M correspond to the different sets of dialogs. The top bars represent confidence intervals at the 95% level.

Outside of motivation and understanding, we also found interactions between age and being a user of Microsoft Internet Explorer for the address book (W_2) and the certificate (W_4) dialogs. This was expected, as these users have likely encountered these dialogs before. In the address book dialog, users of Internet Explorer were more likely to pick the safest response, while in the certificate dialog (W_4), the opposite relation held.

4.4 Discussion

One of the primary goals of this study was to show differentiated results for low- and high-risk conditions to demonstrate that our redesigned dialogs improved participants' abilities to make appropriate security choices in each of the conditions. However, our results did not show differentiated motivation and safe response improvements for the low- and high-risk conditions. For both dialogs presented in low- and high-risk conditions (W_1 and W_2) we found that in the majority of cases motivation and safe response did not significantly increase for the redesigned dialogs in either condition. It is likely that the redesigned dialogs were not more effective than existing dialogs and were not able to increase motivation or safe response in either case. It is also possible that the high security-priming scenarios that were used to prompt the high-risk condition were poorly designed and did not prompt a high-risk response. However, this is less likely as 3 out of 8 had significantly higher levels of motivation and safe response for the high-risk condition. Further research is needed to better determine how users respond to high- and low-risk conditions and how to consistently design better security dialogs.

One of our redesigned dialogs, the M version of the address book dialog (W_2), turned out to be particularly ineffective. It decreased participants' understanding, increased user motivation and safe response in the low-risk condition, and did not increase motivation or safe response in

		E	G	M	E vs. G				E vs. M			
					c	*s.e.*	*z*	*p*	*c*	*s.e.*	*z*	*p*
W_1	S_1	59%	56%	49%	-0.098	0.259	-0.378	.7054	-0.382	0.253	-1.513	.1303
	S_2	79%	88%	95%	0.712	0.381	1.870	.0610	1.702	0.510	3.334	**.001**
W_2	S_1	37%	50%	79%	0.516	0.272	1.898	.0576	1.819	0.313	5.819	**<.0001**
	S_2	88%	91%	93%	0.333	0.425	0.783	.4340	0.541	0.437	1.237	.2160
W_3	S_1	32%	56%	48%	0.982	0.261	3.761	**<.0001**	0.684	0.271	2.523	**.0120**
	S_2	62%	81%	74%	0.942	0.306	3.081	**.002**	0.559	0.3000	1.865	.0620
W_4	S_1	64%	89%	91%	1.490	0.369	4.040	**<.0001**	1.696	0.377	4.495	**<.0001**
	S_2	57%	81%	93%	1.166	0.288	4.053	**<.001**	2.268	0.410	5.530	**<.001**

Table 4.7: Results of two logistic regressions comparing safe response levels between dialog sets. The two leftmost columns show the dialog (W_1 through W_4) and the scenario (S_1 and S_2) shown to the user. The next three columns show the proportion of participants who said they would pick the safe option, per dialog set (E stands for Existing, G for Guidelines-based, and M for mental-model-based). The next four columns show the parameters of the regression comparing the Existing and the Guidelines-based sets, and the rightmost 4 columns show the parameters of the comparison between the Existing and the Mental-model based sets. **c** is coefficient, **s.e.** is standard error, **z** is z-value, and **p** is p-value.

the high-risk condition. One potential explanation for this unexpected behavior is the amount of information that version contained: the existing version had 44 words and 4 options, and the guidelines-based version had 40 words and 3 options, while the mental-model-based version had 163 words and 6 options. The extra text included the results of an anti-virus scan, and an explanation of the consequences for each option. The large amount of information may have undermined participants' abilities to understand (or motivation to read) the redesigned dialog, or some element of the added text might have confused them.

Although our redesigned dialogs appear not to help participants differentiate between high- and low-risk conditions, we were able to demonstrate that it is possible to use a relatively simple redesign process to improve some security dialogs for high-risk conditions. Beyond the importance of testing whether participants could differentiate between high- and low-risk conditions, it was also important to show that our results were applicable across different types of contextual scenarios. To do so, we presented participants with low and high security-priming contexts (S_1 and S_2). Further work is necessary to determine which aspects of the redesigns contributed to the successful increases in motivation and safe response and which aspects were not successful at increasing understanding, motivation or safe response. In Chapter 5 I present a methodology aimed at solving these problems.

4.4.1 Limitations

Our study had a variety of limitations. First, the study is based on self-reported survey data, and as such it may not reflect what users would do when confronted with dialogs during their regular computer use. Also, literature suggests that habituation should be considered when studying dialogs [97]. To the best of my knowledge, repeated, long-term exposure to computer security dialogs has not been studied before, in part because of the difficulties in setting up adequate experimental designs. However, a deeper look at the answers of our participants show that only a small proportion of them reported that they ignored our dialogs, either because they had seen them before or for other reasons. If our participants had been habituated to our set of existing dialogs,

	W1: Encryption warning	W2: Address book warning	W3: Attachment warning	W4: Certificate warning
Understanding	c=0.2693; se=0.3743; z=0.720; p=0.4718	**c=1.7099; se=0.4317; z=3.961; p=7.46e-05**	**c=0.7567; se=0.1911; z=3.959; p=7.53e-05**	**c=0.4945; se=0.2277; z=2.172; p=0.0298**
Motivation	**c=0.9021; se=0.3670; z=2.458; p=0.0140**	**c=1.4442; se=0.4158; z=3.473; p=0.00051**	**c=1.6107; se=0.1751; z=9.195; p<2e-16**	**c=1.6113; se=0.2125; z=7.582; p=3.41e-14**
Gender	c=-0.4687; se=0.3420; z=-1.370; p=0.1706	c=-0.0371; se=0.4255; z=-0.087; p=0.9304	c=-0.2056; se=0.1799; z=-1.143; p=0.2532	c=-0.1445; se=0.2096; z=-0.690; p=0.4904
Age	c=-0.0045; se=0.0223; z=-0.204; p=0.8384	c=0.0724; se=0.0451; z=1.606; p=0.1083	**c=0.0347; se=0.0150; z=2.313; p=0.0207**	c=0.0003; se=0.0156; z=0.025; p=0.9802
Use of Internet Explorer	c=0.6684; se=1.0370; z=0.645; p=0.5192	**c=2.8604; se=1.4160; z=2.020; p=0.0433**	c=-0.2554; se=0.5716; z=-0.447; p=0.6549	**c=-1.7783; se=0.6596; z=-2.696; p=0.0070**
Use of Internet Explorer – Age	c=-0.0024; se=0.0296; z=-0.082; p=0.9347	c=-0.0837; se=0.0501; z=-1.670; p=0.0948	c=-0.0035; se=0.0177; z=-0.200; p=0.8414	**c=0.0536; se=0.0205; z=2.608; p=0.0091**

Table 4.8: Logistic regression coefficients of interactions between variables (H4), per dialog. Dark cells show significant, positive values, and grey cells show significant negative values.

we would expect to have seen a higher number of people ignoring them. Another factor that might have affected participants' response is the novelty of redesigned dialogs. Although we found no evidence in this direction, this remains a limitation of our study.

Another confounding factor might be the possible learning process that takes place after repeated exposures to the same set of questions with different dialogs. A technical limitation of the software we used to implement the survey (`http://www.surveygizmo.com`) prevented us from tracking the random order in which participants saw our dialogs. Although randomization might counter-balance learning effects, we acknowledge that this does not necessarily cancel out the effects. One improvement to the experimental design would be to show a single dialog to each participant. We decided to show five dialogs instead of one to reduce the number of participants needed for the study.

Our redesigned sets utilized different layouts of options, longer and more descriptive texts for each option, information about context, and the results of analysis by other tools. However, our experimental design did not allow us to isolate the impact of each of these design changes.

4.4.2 Conclusions

By comparing existing computer security dialogs with two sets of dialogs that we created, we explored relationships between the design of the dialog, understanding of the problem underlying a security dialog, the belief that the problem is important (motivation), the tendency to pick the safest option (safe response), and demographic factors. We found that design changes can lead to improvements in understanding, motivation, and tendency to pick the safest option in some cases, but further work is needed to isolate the impact of various design factors. However, we were unable to help participants differentiate between the appropriate option in high- and low-risk conditions. We also found that, although understanding and motivation are strongly tied to each other, motivation is a slightly more important factor than understanding when it comes to increasing safe response to security dialogs.

Security dialog designers should keep in mind that both the level of importance that users attribute to a dialog and the understanding of the problem underlying a dialog contribute to user response. To be successful, dialogs should both motivate a user to respond, and help users understand the risk, in that order. Future work should look at exactly how much each of these factors, and other factors, contribute to increasing safe response to dialogs.

Chapter 5

A realistic, ecologically-valid observation method applied to the trusted path problem

5.1 Introduction

When asking users to enter credentials, today's desktop operating systems often use windows that provide scant evidence that a trusted path has been established. This evidence would allow a user to know that a request is genuine and that the password will not be read by untrusted principals. In this chapter, we measure the efficacy of web-based attacks that spoof OS windows to trick users of *Mac OS X* (Mac OS) and *Windows Vista/7* (Windows) into entering their device (OS-level) login credentials. The spoofing attacks we tested exploit the fact that these operating systems often request users' device credentials by overlaying windows on top of windows from other principals.

We recruited 504 users of Amazon's Mechanical Turk to evaluate a series of games on third-party websites. The third such website indicated that it needed to install software from the publisher that provided the participants' operating system: Microsoft's Silverlight for Windows Vista/7 users and Apple's QuickTime for Mac OS users. The website then displayed a spoofed replica of a window the participant's client operating system would use to request a user's device credentials. In our most effective attacks, over 20% of participants entered passwords that they later admitted were the genuine credentials used to login to their devices. Even among those who declined to enter their credentials, many participants were oblivious to the spoofing attack. Participants were more likely to confirm that they were worried about the consequences of installing software from a legitimate source than to report that they thought the credential-entry window might have appeared as a result of an attempt to steal their password.

Stealing credentials via spoofing requires two separate forms of social engineering. First, the attacker must give the user *motivation* to enter her credentials. This means creating a task or scenario that the user wants to complete, or believes she needs to complete, and that can only be completed if she provides her credentials. In the extreme, this could be as simple as presenting a

This chapter is largely a reproduction of a paper co-authored with Lorrie Cranor, Julie Downs, Saranga Komanduri, Stuart Schechter, and Manya Sleeper [15].

window that can be dismissed only by entering credentials, and that the user will want to close. Second, the attacker must spoof the credential-entry interface in place of a genuine interface with sufficient fidelity that the user will *trust* it.

In our attacks, we tried to motivate users to enter credentials by convincing them that they should install software, and that their credentials were required to do so. In many contexts, credentials are indeed required to install software. On Mac OS, installing software requires a privilege elevation, which causes a credential-entry window to appear in the center of the screen. On Windows, User Account Control (UAC) [20] is a protection mechanism that controls the elevation of privilege to allow installations and other sensitive actions. As with Mac OS, it uses a window to verify user intent before elevating. However, unlike Mac OS, its default configuration will only request a username and password if the current user does not have the administrator privilege required for elevation. Furthermore, Windows attempts to differentiate UAC windows from other windows by dimming the contents of the rest of the screen when a UAC window is present.

Windows applications may also request that the OS present the Windows credential-entry experience (CredUI) [48] in situations where the user should re-authenticate. Notably, Microsoft Outlook[1] may request the users' credentials via CredUI, and Microsoft Lync[2] may request credentials through a custom interface, even if the user has already provided these same credentials to login to her PC. CredUI does not dim the portions of the users' screen controlled by other principals.

On Mac OS, we spoofed the password-entry window that appears when users need to elevate privileges to authorize software installs (Figure 5.1c). For Windows users, we tested both a spoofed User Account Control [20] window (Figure 5.1a) and a Windows CredUI [48] window (Figure 5.1b).

Under our experimental conditions, we found that in over 20% of trials, our most effective attacks yielded passwords that users later admitted that were their genuine device login passwords. Furthermore, when participants who refrained from revealing their credentials were asked why they did so, only 35.3% confirmed that they thought the password-entry interface might be an attempt to steal their passwords (a lack of trust in the interface). Many of the remaining participants may have oblivious to the attack, but simply weren't motivated to enter their credentials to install new software.

The consequences of an attacker obtaining device credentials are different than, and in some cases may be more severe than, an attacker gaining website credentials. They also vary greatly with the security posture of the victim or the victim's organization. If the user's device allows remote access and no firewalls block it, an attacker with the username and password will obtain complete control of the user's device account and will be able to install backdoors and key loggers. If the compromised user has administrative access to the machine, the attacker's keylogger can collect usernames and passwords from other users and the attack can snowball. Enterprise users may be more likely to have remote access restricted by firewalls, but in many cases a Windows domain username and password may be sufficient to allow a new computer to be added to a domain, allowing an attacker to bypass firewalls. Even if compromised credentials cannot be used for remote device access, many enterprises also allow access to web- and smartphone-based email clients using these username and password credentials.

[1] `http://office.microsoft.com/en-us/outlook/`

[2] `http://office.microsoft.com/en-001/lync/`

(a) UAC treatments.

(b) CredUI treatments

(c) MacOS treatments.

Figure 5.1: Our spoofed credential-entry windows

5.2 Experimental design

We designed our experiment to determine the fraction of users who would enter their passwords into a spoofed OS window, the fraction who would detect that the window was spoofed, and what clues (if any) users looked for to detect if a window was spoofed.

Our experiment mimicked the experience of going to a previously unvisited website and re-

ceiving a spoofed installation window designed to steal username and password credentials. The rules of informed consent dictate that we disclose the identity of our research institution before collecting data from users. We thus needed a study design that would allow us to minimize the likelihood that participants' trust in our research institution would cause them to behave less safely than they otherwise would.

We received Institutional Review Board (IRB) approval to create a deception study in which participants were told that their job was to help us evaluate third-party gaming websites. We did so because any trust participants had in our institution should not extend to third-party websites, and so before directing participants to these sites we notified participants that these sites were outside our control. However, the simulated attacks took place on a *confederate* gaming website, which we secretly did control. We use the term *confederate*, because this website was used in much the same way human confederates are employed in other human subjects experiments: presenting the illusion of being a third-party outside the influence of the researchers, creating a situation to which participants could respond (the spoofing attack), and recording participants' behaviors in situations in which participants may have believed themselves to be free from researchers' observation.

5.2.1 Recruitment and screening

Participants volunteered for a Human Intelligence Task (HIT) we had posted on Amazon's Mechanical Turk, as detailed in Appendix C.1. We asked 504 workers to evaluate a series of three online games, hosted on third-party websites, for such factors as level of enjoyment and age appropriateness. The third such website, controlled by us, spoofed an OS credential-entry window requiring device-login credentials.

Participants had to be at least 18 years old, in the United States while taking the survey, and have an MTurk approval rating of at least 95%. Participants must not have participated in any previous similar study from our lab, including earlier versions (pilots) of this experiment. We used browser-submitted HTTP headers to verify that participants were using either Windows Vista/7 or Mac OS X or higher. We paid $1.00 to each participant who qualified for and completed our survey.

5.2.2 Tasks

Upon accepting the HIT, we redirected participants to a survey site operated in the domain of our research institution. Our survey requested that the participant maximize the browser window so that, when participants reached the third-party website conducting the simulated attack, the appearance of the spoofed window within the browser content area would be less suspicious. We posited that an attacker could use social engineering to convince most users to maximize their browsers.

Our survey presented three game evaluation forms in sequence, each of which asked participants to click on a link opening a third-party online gaming website (a different one for each form), play an online game, and then answer questions about the game. Each form had a link to the game, and participants were instructed to open the link while pressing the ctrl (Windows) or command (Mac OS) key, opening a new browser tab. Participants were asked to play each game for two to three minutes, then close the browser tab and return to the survey form. The evaluation

form, given in Appendix C.2, asked about the game's enjoyability and age-appropriateness, and gave participants the option to report being unable to play a game.

The first two games were real games operated by real gaming websites outside of our control. These two tasks were included to add legitimacy to the subterfuge that we were indeed soliciting evaluations of third-party games. The third and final website, `yourgamefactory.net`, was the confederate website that we secretly operated. It was at this website that participants were told that they needed to install new software to play the game and then saw a spoofed credential-entry window. The exact message of what would be installed and the layout of the spoofed windows was determined by the participant's treatment group.

If participants entered their credentials into the confederate website, they encountered a page that explained that the game had been removed. Regardless, once participants completed evaluating the game, which simply required them to check a box indicating that they couldn't or wouldn't play it, they were presented with our exit survey (see Appendix C.3).

5.2.3 Exit survey

The biggest methodological challenge we faced was to determine if a participant's credentials would have been compromised by our simulated attack. In other words, had they typed their genuine device username and password into the credential-entry window?

Even if it were ethical to send the passwords to our servers without participants' consent, there would be no sure way to tell if the password was legitimate without trying to gain access to the participants' computers—a criminal act. Simply asking participants to tell us if they had entered their genuine credentials seemed likely to result in underreporting. Upon discovering our deception, many would surely be both tempted to deceive us in their response. Furthermore, participants might have legitimately feared that we, as perpetrators of a deception, might misuse their credentials. If they feared their credentials might still be abused, they might justifiably have believed their safest action was to deny that the credentials they revealed were indeed genuine.

While we did not immediately send the contents of the password field to our server, we did send its length l. If the field was non-empty ($l > 0$), we used this value in the exit interview when asking the following 'yes' or 'no' question:

> *You entered l characters into the password field of the dialog box. This password you entered is stored in your browser but we have not sent your password to our servers. Was the password you entered a real password for an account on your computer?*
>
> ◯ *Yes, I did enter a genuine password (we'll immediately delete any records of the password we kept in your browser)*
>
> ◯ *No, I did not enter a genuine password*

If the answer was no, we used a follow-up question to try to identify participants who had entered their genuine passwords but had decided to deny doing so.

> *Since you did not enter a genuine password into the password field of the dialog box, may we collect the contents of this field for analysis?*

◯ *Yes, since the password I entered is not a genuine password you may send it to your servers.*

◯ *No, the password I entered was actually a genuine password. Please immediately delete any records of it in my browser and do not send it to your servers for analysis.*

◯ *No, I have another reason for not wanting the password I entered sent to your servers (please explain)*

We designed this question so that if participants had entered genuine credentials, they would believe the safest option was to admit to doing so. If participants did not enter their genuine credentials, they would only need to enter an arbitrarily short explanation to keep the values they entered private. Participants who admitted that their password was genuine in either of the above two questions would be considered to be *compromised*.

If a participant did not enter a genuine password, we first asked them to use freeform text to "please explain why you didn't enter your password into the password-entry window." We then asked participants to "please indicate which (if any) of the following factors contributed to your decision to not enter your password." Participants could check any number of options, the order of which was randomized for each participant. The options included legitimate concerns, such as not wanting to take the time to install software. The options also included items that should not have been concerns, such as fearing that updated software, published by the company that provided their client OS, would harm their computers. The combination of both types of options was used so that participants would have to think about whether each option really did match their set of concerns. Amongst the list of options was one that stated "I thought that the password-entry window was trying to steal my password." Participants who checked this option are considered *wise* to the spoofing attack. Those who did not are considered oblivious to the spoofing, or *oblivious* for short.

Only after these questions did we disclose the deception and explain the purpose of the study. After doing so, we asked participants such questions as whether they suspected that the window was spoofed and whether they took any actions to test if the window was authentic. At the end of the survey we asked additional demographic questions.

5.2.4 Instrumentation

We instrumented the confederate gaming website to record participants' OS type, browser client name and version, screen size, browser viewport size, and the position of the top left corner of browser's viewport relative to the top left corner of the screen. We also recorded participants' mouse movements, clicks within the page content (which included our spoofed window), and number of keystrokes in the username and password field, as well as the timing of each of these events measured in milliseconds.

If the participant tried to submit credentials using the spoofed credential window, we encrypted the contents of the username and password fields using a random one-time-use symmetric encryption key retrieved from, and stored exclusively by, our servers. We stored the ciphertext of the username and password in the client's browser local storage, but discarded the key from the client so that the client could not decrypt the ciphertext. This allowed us to reduce the risk of storing the

#	Treatment	Motivation	Spoofed window
(1)	UAC1	Figure 5.3a	Figure 5.1a
(2)	UAC1-D		
(3)	UAC2	Figure 5.3b	
(4)	UAC2-D		
(5)	CredUI-D*	Figure 5.2	Figure 5.1b
(6)	CredUI		
(7)	CredUI-D		
(8)	MacOS1	Figure 5.3c & Figure 5.4	Figure 5.1c
(9)	MacOS1-D		
(10)	MacOS2	Figure 5.3c	
(11)	MacOS2-D		

Table 5.1: Treatment groups. The credential-entry windows for those treatment groups with a suffix of '-D' had no cancel button or had a disabled cancel button, as well as the window close box disabled.

contents of the username and password fields in the participants's browser, yet gave us the option to transmit the information to the server later if we obtained the participant's consent.

5.2.5 Implementing spoofed windows

We spoofed the OS window using HTML, CSS, and Java-script. We did not use Flash or other plugins. The window could be moved, though only within the browser content region. For all Windows treatments, we rendered the spoofed window using the default Windows color scheme and implemented a translucent chrome. We also used the same fonts as the genuine Windows dialogs, though because they were rendered on the client, the browser's zoom level could cause them to be rendered the wrong size—a problem that could have been solved by rendering text on the server. Similarly, on Mac OS we tried to match the OS fonts, colors, and other aspects of the genuine window's appearance.

Genuine credential-entry windows often have the username field pre-filled. Since we simulated attacks that did not have access to the username, these fields were left blank in all treatments. We used the generic 'flower' icon to represent the user account in the UAC dialog, as, if the user had a user-specific icon to represent their account, the attacker would not have access to it.

5.2.6 Treatment groups

Table 5.1 shows the 11 treatment groups in our experiment. Participants using a browser on Windows were randomly assigned one of 7 treatments: 4 UAC treatments and 3 CredUI treatments. Participants using a browser on Mac OS were randomly assigned to one of 4 MacOS treatments.

UAC

In the UAC1 treatment, the content of the webpage above which the credential-entry window appeared stated that the game required Microsoft Silverlight (Figure 5.3a). After four seconds a

Figure 5.2: Web content shown for the CredUI and CredUI-D*treatments.

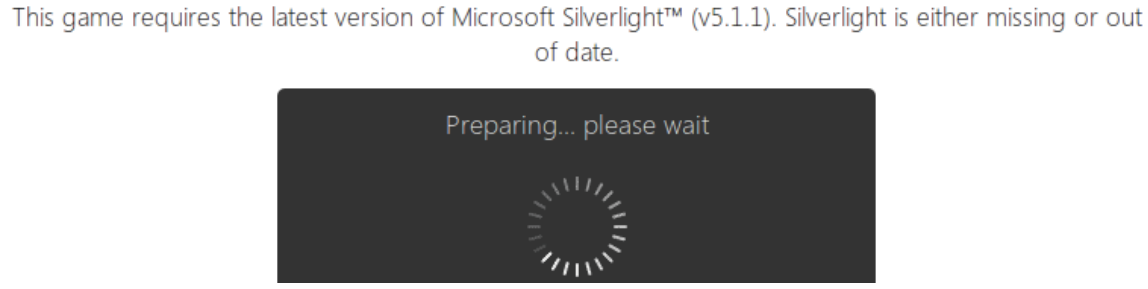
spoofed UAC window appeared asking the participant to consent to the installation of Silverlight and to provide credentials to do so, as illustrated in Figure 5.1a.

The UAC2 treatment was identical except that the installation window instructed participants to "verify that the publisher of this application is Microsoft before installing it" (see Figure 5.3b). If users were to check the spoofed window, they would be reassured that Microsoft was indeed listed as the publisher. We hypothesized that users would focus on whether the publisher listed was Microsoft, be reassured that the software was not a threat when they verified that publisher field matched (and was the publisher of their OS), and would thus be less likely to notice that the window in which the publisher field appeared was, in fact, the real threat.

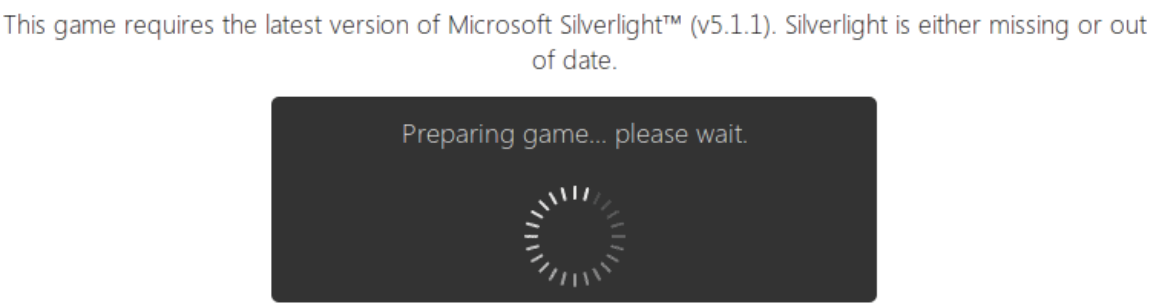
Genuine UAC windows are presented on top of a dimmed desktop and, when credentials are required, the user name is often already filled in. We opted not to dim the region of the screen under our control – the web content region – as we believed that dimming a limited portion of the screen would raise more suspicion than it would dispel.

CredUI

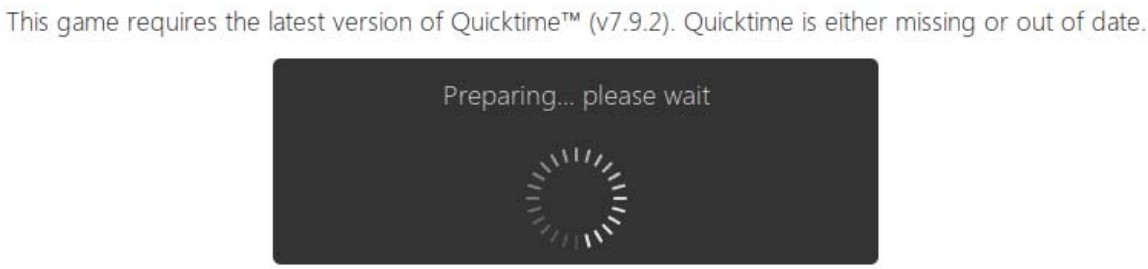
In the CredUI treatment, a webpage appeared explaining that the game was being prepared (see Figure 5.2) and, after four seconds, the spoofed CredUI window illustrated in Figure 5.1b was overlaid above it. We placed the name of the browser (e.g. "Internet Explorer" or "Chrome") into the heading of the window to represent the principal requesting the password. Because we did not have a user name in our experiments, the field was left blank and a field label, "Username," was overlaid on top as it would be in the genuine window. (We later discovered that real Windows



(a) Variation for UAC1 treatments.



(b) Variation for UAC2 treatments.



(c) Variation for MacOS treatments.

Figure 5.3: The contents of the confederate website's pre-installation page, over which the credential-entry window appears, were slightly different for each treatment group. Figure 5.2 shows the full page, whereas the remaining figures show variations used for other treatments.

interfaces actually use the two-word phrase "User name." This small error was common to all Windows treatments.)

MacOS

With Mac OS, we simulated the installation of QuickTime as this software is also a common browser add-on and, like Silverlight on Windows, it would be signed by the same company that provided the user's OS. The sequence of events that start an installation on Mac OS begins with a

Figure 5.4: Spoofed installation-description dialog used exclusively in the MacOS1 treatments.

dialog that describes the software to be installed, followed by a request for credentials. Thus, we first showed the webpage that triggered the fake installation (Figure 5.3c) and, four seconds later, showed a spoofed installation-description dialog (Figure 5.4). Only after the participant clicked to continue with the installation did we show the spoofed password-entry window (Figure 5.1c). Simulating the installation-description dialog may add realism for Mac OS users who expect to see it before entering their password. However, it also presents the user with an additional opportunity to cancel the installation sequence before the credential-entry window appears. For this reason we omitted several steps that users typically see during the Mac OS installation process, such as selecting the filesystem location in which the application will be stored and consenting to a license agreement.

In the MacOS2 treatment, we did away with the spoofed installation-description window step.

Mac OS puts labels to the left of text fields, rather than overlaying them within, so the username field was completely empty. Mac OS also centers credential-entry windows on the screen—a feature that we did not replicate. We failed to capitalize the 'T' in QuickTime on the "preparing game" webpage, but capitalized it correctly in the spoofed installation-description dialog.

Close-disabled variants

In a pilot of a spoofed UAC window we mistakenly activated the 'Yes' button, which submits the credentials, without waiting for the user to enter characters into the username and password field. Prior research suggests [66] that users will often ignore most of a dialog content and jump straight to its options. Many participants saw the "Yes" button and pressed it without entering any credentials. From this early mistake, we hypothesized that participants might be more likely to enter their credentials if the option to dismiss the spoofed window was deactivated.

We paired each of the above treatments with a second treatment in which the 'cancel' or 'no' button was removed and the window close box (the 'X' at the top right in Windows) was disabled.

	CredUI-D*		CredUI		UAC1		UAC2		MacOS1		MacOS2	
	~~Close~~		Close	~~Close~~	Close	~~Close~~	Close	~~Close~~	Close	~~Close~~	Close	~~Close~~
compromised	9 (19.1%)	11 (20.8%)	6 (15%)	15 (27.3%)	17 (34.7%)	10 (20.8%)	15 (31.9%)	4 (7.8%)	4 (7.7%)	3 (6.2%)	2 (4.3%)	
wise	18 (38.3%)	18 (34%)	16 (40%)	15 (27.3%)	14 (28.6%)	18 (37.5%)	14 (29.8%)	15 (29.4%)	10 (19.2%)	14 (29.2%)	26 (56.5%)	
oblivious	19 (40.4%)	24 (45.3%)	18 (45%)	22 (40%)	18 (36.7%)	18 (37.5%)	18 (38.3%)	22 (43.1%)	22 (42.3%)	31 (64.6%)	18 (39.1%)	
unexposed	1 (2.1%)	0 (0%)	0 (0%)	3 (5.5%)	0 (0%)	2 (4.2%)	0 (0%)	10 (19.6%)	16 (30.8%)	0 (0%)	0 (0%)	
Total	47 (100%)	53 (100%)	40 (100%)	55 (100%)	49 (100%)	48 (100%)	47 (100%)	51 (100%)	52 (100%)	48 (100%)	46 (100%)	

Table 5.2: Disaggregated data for the attack rates, per condition.

These treatments are labeled with the suffix '-D,' and are otherwise identical to their suffixless counterparts.

CredUI-D*

The final treatment, CredUI-D*, was exactly the same as CredUI-D, except that participants were not asked to maximize their browsers at the start of the study. We included this treatment to determine if it was really necessary for attackers who wanted to spoof an OS window to first trick users into maximizing their browser windows.

5.3 Results

Our results reveal that spoofing attacks are effective, with 18% of participants over all treatment groups providing what they would later admit were valid login credentials for their device. These participants were classified as *compromised*. For reasons discussed in Section 5.3.2, 6% of participants were *unexposed*, not seeing the spoofed password-entry window.

We asked those participants who did not enter a password to indicate which, if any, factors contributed to their decision not to enter their password. As shown in Figure 5.6, 48% of those we asked reported that they thought the password-entry window was trying to steal their passwords, and we classified them as *wise* to the attack. 36% did not provide this factor and were classified as *oblivious*.

5.3.1 Participants

We ran our experiment between January 25 and February 5 2012, and collected results for a total of 504 participants. Not included are an additional 28 participants who were recruited but never visited the confederate gaming website. We also did not include 15 participants who, despite our instructions, responded to the survey from outside the United States.

To identify participants who may not have made conscientious attempts to read and answer questions, we included a multiple-choice question that any participant should have been able to answer correctly: "The power switch on a computer is used to ___?" This approach was inspired by Downs et al. [30]. All but two participants answered this question correctly, suggesting that the majority of participants in our results made a conscientious attempt to read and answer our survey.

Participants were 28 years old (σ =9.6 years) in average, 55% were male, and 78% were caucasian. The top two reported occupations were 'student' (33%) and 'unemployed' (13%). To gauge their level of expertise, we asked five technical questions on topics such as encryption and

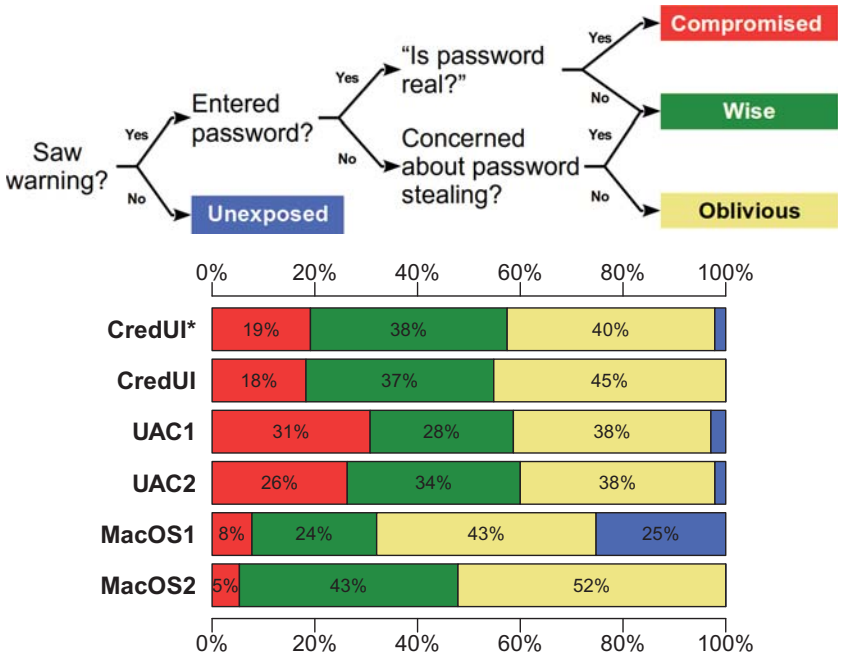


Figure 5.5: Attack efficacy. *Compromised* participants entered username and password and later admitted that these were valid login credentials for their device. Participants *wise* to the attack didn't admit to entering valid login credentials, and checked the box labeled "I thought that the password-entry window was trying to steal my password." Participants categorized as 'oblivious' were not wise to the attack, but did not fall victim because they had other reasons for not wanting to enter their passwords. *Unexposed* participants may have not seen the credential-entry window (e.g. because they closed the install window presented prior to the credential-entry window in condition MacOS1). With the exception of CredUI-D*, all treatment groups shown above correspond to the merge of the corresponding cancel-enabled and cancel-disabled pair. Disaggregated data can be found in Table 5.2. The choice of treatment, as shown above, had a statistically significant effect on the outcome.

web security. No participant answered all questions correctly, 11% were able to answer correctly 4 out of 5 questions, and 54% answered one or no questions correctly. 28% reported knowledge of at least one computer programming language. Finally, participants took an average of 17 min 23 secs (σ =18 min 15 secs) to complete the study.

5.3.2 Attack efficacy

Of the pool of 504 participants in all treatment groups of our experiment, 142 (29%) entered one or more characters into a spoofed password field. Recall that we used two separate questions to try to compel participants to admit the truth if they had entered their genuine credentials. Of the 142 participants who entered a value into the password field, 38 (27%, or 8% of the total participant pool) denied that the text that they had typed was their genuine device password and, in response to the second question, consented for us to see what they had typed. Another 9 (6% of the participants who typed a password, or 2% of the total participant pool) denied that the password was genuine, but refused to consent for us to see it. The remaining 95 participants (67% of participants who typed a password, or 18% of the total participant pool) admitted to typing

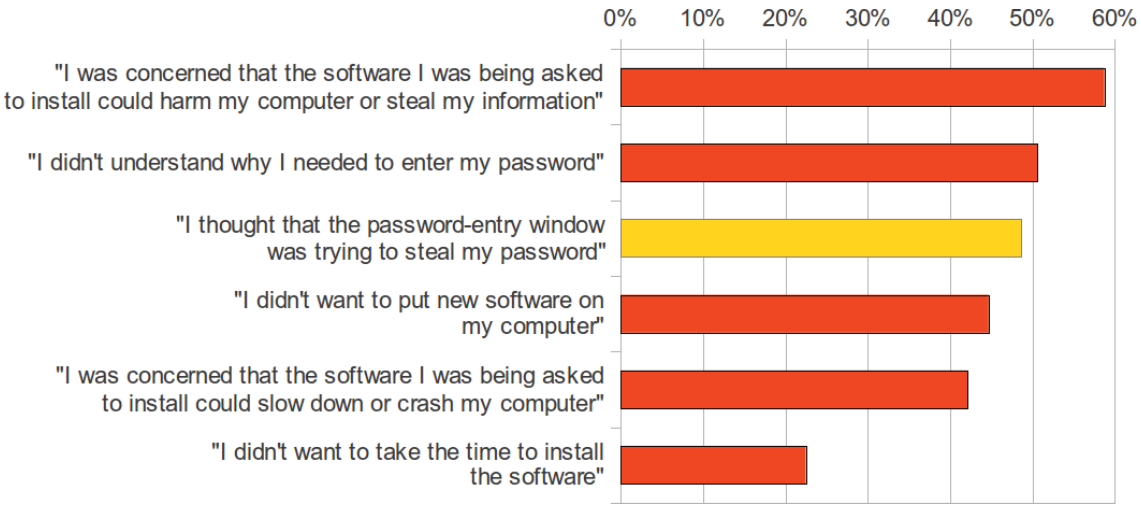


Figure 5.6: Participants who did not enter any passwords were asked the following multiple-answer question: "Please indicate which (if any) of the following factors contributed to your decision to not enter your password." Participants who picked the third option from top were categorized as 'wise' to the attack; the others were categorized as 'oblivious'.

their genuine credentials, and all but one did so by answering 'yes' to first of the two questions. (Note, however, that participants may have revised their answer to the first question after seeing the second question.)

The results of each attack are presented in Figure 5.5, and the disaggregated numbers are presented in Table 5.2. Our research questions led us to two hypothesis tests, one to determine if close-disabled treatments performed better than their close-enabled peers and one to determine if browser-maximization had an impact on our results. In our analysis we use χ^2 tests to indicate the statistical likelihood that differences between treatment groups were the result of chance. We applied eight tests, and corrected for multiple testing using the Bonferroni method, assuming $\alpha = .00625$ in place of $\alpha = .05$.

In both UAC treatments, the close-disabled treatment pair had a higher compromise rate than a close-enabled treatment pair. In aggregate, the five paired close-disabled treatments had compromise rates (43/215, 20%) slightly higher than their close-enabled peers (43/237, 18%). However, the differences in attack efficacy between conditions with enabled and disabled cancel buttons were not statistically significant ($\chi^2(4) = 1.34$, $p = 0.855$).

To highlight changes between the fundamental designs, we merge close-enabled and close-disabled treatment groups together in Figure 5.5 (disabling close boxes did not appear to have a significant effect, and was applied with equal frequency to each of the fundamental designs). Over the remaining six treatment groups, we found that the choice of treatment group had a significant effect (after correcting for multiple testing) on both the fraction of participants categorized as *compromised* ($\chi^2(5) = 31.13$, $p < 0.001$) and the fraction categorized as *wise* ($\chi^2(5) = 24.71$, $p < 0.001$).

Some participants were *unexposed* to the spoofed password-entry window because they closed the browser tab before the credential-entry window could appear. Those in the MacOS1 treatments may have aborted installation when the spoofed installation-description dialog was displayed (Fig-

ure 5.4), which came before the credential-entry step.

In every group, at least one participant provided credentials that they later admitted were genuine. For the Windows treatments, the compromise rates in Figure 5.5 fall between 15% and 35%, with the maximum compromise rate only $2.31\times$ greater than the minimum—a relatively tight range given the sample sizes involved. When combined, participants in all the UAC attack treatment groups had a higher compromise rate (57/192, 29.7%) than those in the two CredUI attack treatments (17/93, 18.3%), though not significant when correcting for multiple testing ($\chi^2(1) = 4.24$, $p = 0.039$).

Many participants who were not compromised may still have been fooled, trusting that the credential-entry window was genuine but failing to enter their credentials due to a lack of motivation. More participants refused to enter their credentials because they were concerned that "the software I was being asked to install could harm my computer or steal my information" (209/356, 58.7%) than because they were afraid to enter their credentials (173/356, 48.6%). Similarly, more participants withheld their credentials not because they detected the spoofing attack, but because they were not motivated to install new software (80/356, 22.5%). Furthermore, a relatively high proportion of participants across all treatments (180/356, 50.6%) volunteered that they did not enter their credentials because they did not know what credentials to enter.

A greater proportion of participants in the two CredUI treatments were wise to the attack (51/93, 54.8%) than those in the four UAC treatments (67/192, 34.8%, $\chi^2(1) = 10.27$, $p = 0.001$, significant when corrected for multiple testing.)

A disproportionately small fraction of participants were compromised by the MacOS2 treatments (5/94, 5.3%), which did not spoof an installation-description window before the credential-entry window, and a disproportionately large fraction were wise to it (40/94, 42.6%). In contrast, the MacOS1 treatments did present an installation-description window. This presented participants who did not want to install QuickTime an opportunity to abandon the process before seeing the credential-entry window, so 25% of participants were not exposed to it (hence *unexposed*). Even accounting for those unexposed, more participants were compromised by the treatments that included the spoofed installation-description window (8/103, 7.8%) than without it (5/94, 5.3%).

While the aggregate compromise rates for participants in the Mac OS treatments were lower than for the Windows treatments, it would be premature to conclude that Mac OS users are less vulnerable to spoofing; we put more effort into tuning our Windows attacks and the treatments targeted different software to install (Silverlight vs. QuickTime). Even when we did present an installation-description dialog, we skipped a number of steps in the installation ritual.

The difference between the rates at which participants in the Mac OS treatments were wise to spoofing (65/197, 33.0%) and those in the Windows UAC treatment groups (61/199, 30.7%) were well within the margin of error ($\chi^2(1) = 0.1539$, $p = 0.6948$).

The CredUI-D* treatment, in which participants were not asked to maximize the browser window at the start of the survey, did no worse than the identical treatment in which participants were asked to do so (CredUI-D). Of those in the CredUI-D* treatment, our instrumentation indicates that 78% already had their browser sized to consume at least 80% of the screen's area. It's possible that convincing users to maximize browser windows may raise more suspicions than it dispels. In future experiments we may consider removing the window-maximization step, though we cannot say with confidence that this will increase attack efficacy. UAC windows are bigger than CredUI windows, and, therefore, failing to maximize a small browser window might be more likely to

cause a user to become wise to a UAC-based attack than a CredUI-based attack. While we believe that convincing users to maximize their browser windows in advance of an attack poses little challenge to social engineers, our data suggests that doing so may be unnecessary.

Finally, after we disclosed the deception to our participants, we asked them whether they "know that the password-entry window was actually mimicked by the website, and not a real password request from [their] operating system." The answers to this question ranged from 'I was completely sure that the password entry wasn't real' to 'I never suspected'. While no 'compromised' participants reported to be completely sure about the deception, between 16% and 25% of participants who were not 'compromised' reported that they were sure of the deception.

5.3.3 Drop-out rates

During the study, 136 participants began answering the survey but dropped out before finishing. We have only partial information about who these participants were and what they did, but our instrumentation allowed us to know at what point they dropped out of the study. Out of the 136 participants who did not complete the study, 6 were from outside the United States and would not have been included if they finished. 18 participants dropped during the consent form (that is, they did not reach the first game), 56 dropped during the first game (they did not see the second game), 23 dropped during the second game, and 33 dropped during the third game. Out of these 33, two participants skipped all the games (they checked the "I could not play this game" checkboxes), 25 evaluated two games and dropped during the third game (we don't have evidence that these participants visited the gaming website or saw our spoofed dialog), and finally 6 participants reached the third game, saw our simulated dialog and interacted with it. Two of them did not enter a password, returned to our survey, answered a few more questions, and then dropped out; three of them entered a password and never returned to the survey; and finally one last participant entered a password, returned to the survey, said that his or her password was real, and then dropped out. We included these last 6 participants in our analyses reported above.

5.3.4 Reasons for suspecting spoofing

We asked participants with suspicions to explain them to us. For many participants, especially Mac OS users, the empty username field was a source of suspicion. If an attacker can obtain the user's username, the credibility of that spoofed window, and the compromise rate, would likely rise.

Some participants found ways to test whether the spoofed credential-entry window was real, sometimes using techniques we had not anticipated. One Mac OS participant used Exposé, which reveals the set of OS-level windows, to see that the credential-entry window was only present within the browser. Other Mac OS participants noticed that we had failed to center the spoofed dialog on their screens and knew that the genuine dialog is centered. Some participants in the Windows UAC treatment group were familiar with UAC and knew their computers were configured such that they would not request a password to complete an elevation. In the words of a participant, *"when i installed silverlight before it didn't ask for a password."* Similarly, other participants were aware that their account did not have administrator rights. In the words of another participant, *"Microsoft Silverlight is not installed on my computer, and I am not the 'Administra-*

tor' of this computer so I did not know the password. If I had known the password, I probably would have entered it and not thought anything about it."

Removing paths through which users can close a credential-entry window also raised suspicions among some participants on both Mac OS and Windows. For example, a Mac OS participant reported that the spoofed window *"didn't have the clean Apple 'look' [and] didn't have an available cancel button."* Changing the appearance of the window close box and removing the cancel button may have provided a visual clue that the window was fake. Disabling such functionality more subtly (e.g. by making active-looking buttons non-functional) might increase compromise rates, as the only users who would be alerted to the difference would be those who had already decided to close the window (and thus would not be entering their credentials). Some fraction of those users might then decide to enter their credentials if they believed it was the only way to dismiss the window.

Finally, a handful of participants, familiar with research studies, were not fooled by the study scenario and the subterfuge of the fake site. One wrote: *"I'm taking a psychology study. I just figured it was part of the study."*

5.3.5 Follow-up experiment

We performed a second experiment to more tightly bound our estimate of the efficacy of one of our most effective attacks: UAC1. We collected data for 199 participants during two solicitation periods on July 20 and 25 of 2012—a fourfold increase in the number of participants per treatment. In both sessions our participant quotas were met within a matter of a few hours, in contrast to our earlier study which was offered over a greater diversity of times-of-day. Participants' demographic data were virtually identical in both experiments. In the second experiment participants were an average of 29 years old with a $\sigma = 9.7$ years (vs. 28 years old, $\sigma = 9.6$), 53% were male (55% in the original), 77% were caucasian (78% in the original), and the top two reported occupations were again 'Student' (28% vs. 33% in the original) and 'Currently Unemployed' (16% vs. 13% in the original). Participants took in average 19 min 57 sec to complete the study with a σ =8 min 26 sec, vs. 17 min 23 sec with a σ =18 min 15 sec.

Out of 199 participants, 52 entered at least one character, and 41 of them (21% of the total) later admitted it was a real password in either the first or the second question described earlier. Of those who did not proceed to enter characters into the password field and were asked why, 63 (32% of the total) indicated password-theft as a concern, causing us to categorize them as *wise*. The remaining 95 (47% of the toal) were deemed *oblivious* to the attack. The most frequently invoked reason for not entering a password was 'concern that the software could damage their computers', checked by 104 participants (52% of the total), and 'not wanting to install new software', checked by 86 participants (43% of the total).

In this follow-up experiment, 24 participants did not complete the survey, 6 of them being from outside the US. Of the 18 remaining, 14 dropped before getting to the third game. Only one of the four remaining participants returned to the survey after having seen the spoofed dialog; we don't have evidence that the other three actually saw the dialog.

The compromise rate in the follow-up experiment (21%) was lower than for the same treatment in the original experiment (27%), although the difference is not significant ($\chi^2(1) = 0.429$, $p = 0.5125$). Figure 5.7 displays the 95% confidence intervals for the compromise rates observed in this experiment.

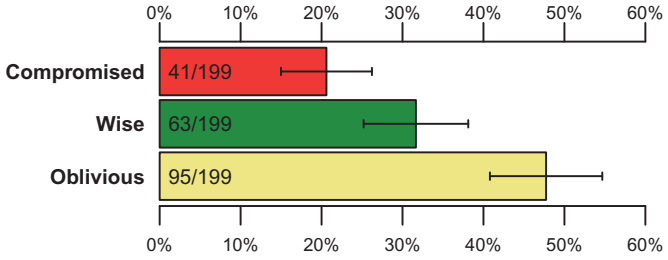


Figure 5.7: Attack efficacy for second experiment, along with 95% confidence intervals. 20.6±5.6% of participants were *compromised*, 31.7±6.5% were *wise* to the attack, and 47.7±6.9% were *oblivious* to the attack.

5.4 Limitations

As with any experiment, our study has limitations that may cause our results to differ from the results of a real attack, including the 5% attack efficacy results for scareware campaigns previously reported by Cova et al. [21, 22].

Our participants were drawn from the population of users of Mechanical Turk who accepted our HIT. This population may differ in important ways from the populations targeted in certain attacks. For example, an attacker targeting a software security company might compromise a smaller proportion of users than were compromised in our experiments, as such individuals may be more likely to detect spoofed windows. Mechanical Turk users may be more or less likely to be using personal (as opposed to a work) accounts and thus the vigilance with which they protect their credentials may differ from the populations targeted in real attacks.

Some factors may have made our simulated attacks more likely to result in a compromise than a real spoofing attack. For example, participants may have recognized the name of our institution in the initial consent disclosure and assumed that researchers would not direct them to an unsafe third-party site. Participants may also have mistyped their credentials but reported that they had entered their valid credentials. Additionally, convincing users to maximize their browser, or doing it for them, may be essential to achieving the compromise rates we saw. However, we did not see evidence of this when we compared the CredUI-D* and CredUI conditions. Attackers may be unable to convince users to maximize windows without causing suspicion.

It is also possible that attackers could achieve compromise rates much higher than those we saw in our experiments. An attacker who could provide a more compelling scenario for entering credentials might be able to compromise many of those users who would not be compromised by our treatments. A real attacker need not repeat mistakes we made when learning to spoof these interfaces, such as using the word ‘Username’ instead of ‘User name’ in Windows and failing to center the Mac OS credential-entry dialog. A real attacker spoofing an installation of Silverlight might put a Silverlight object on the page to detect whether Silverlight was already installed and to identify the current version.

In considering the results of our study, one must also consider that the consequences of a compromise vary widely based on what they are used for. If the user’s device blocks all forms of remote access, the compromise of device credentials may be of no consequence. If the credentials are for the user’s account on an enterprise network, and that enterprise offers remote access to the

network, computing, and services (e.g. email, payroll, etc.), the consequences could be significant. If the user employs the same credentials for other accounts, the consequences may extend even further.

5.5 Conclusion

Only a minority of participants in our study recognized that a spoofed operating system credential-entry window was an attempt to steal their passwords. In our most effective attacks, more than 20% of Windows users entered usernames and passwords into a spoofed UAC window, later admitting that these were genuine device-login credentials. Providing a trusted path through which users can enter credentials securely is not in itself sufficient to prevent such attacks. Rather, operating systems must also forbid the collection of device-login credentials through less secure paths (e.g. spoofable windows), lest users become habituated to entering credentials when straying off the trusted path.

Chapter 6

Attractors: a novel approach to increase computer users' attention to security dialogs

6.1 Introduction

We designed and tested *attractors* for computer security dialogs: user-interface modifications used to draw users' attention to the most important information for making decisions. Some of these modifications were purely visual, while others temporarily inhibited potentially-dangerous behaviors to redirect users' attention to salient information. We conducted two between-subjects experiments to test the effectiveness of the attractors. In these experiments we sent participants to perform a task on what appeared to be a third-party site that required installation of a browser plugin. We presented them with what appeared to be an installation dialog from their operating system. Participants who saw dialogs that employed inhibitive attractors were significantly less likely than those in the control group to ignore clues that installing this software might be harmful.

Reducing the onslaught of interrupting security dialogs might help reduce the strain on users' attention. Some dialogs can be removed by re-architecting systems to reduce the potential for harm, such as by building file parsers in type-safe languages or sandboxing unsafe code. Yet inevitably, some decisions must eventually be made by users. One type of unavoidable decision is the choice to take a risk that some users may embrace and others may reject. For example, some users may want to share their location with an application that others would not share their location with. In other cases, users have knowledge, which the system does not have, that is essential to making a correct choice. For example, the user may know that a particular wireless network is operated by somebody they trust.

Designing user interfaces to facilitate necessary security decisions is especially difficult given that the damage caused by unnecessary decisions has already been done. After years of training to ignore cries of wolf, users are unlikely to become more attentive to them overnight.

This chapter is a partial reproduction of a paper co-authored with Lorrie Cranor, Julie Downs, Saranga Komanduri, Rob Reeder, Stuart Schechter, and Manya Sleeper [14]. Most of this work was performed while interning at Microsoft Research.

Figure 6.1: Installation dialog used as Control. Only the suspicious publisher ('Miicr0s0ft.com') is shown.

From this unfortunate starting point, we set out to test user interface elements designed for attracting users' attention to critical information in a security-decision dialog. We call these user interface elements *attractors*.

We conducted two between-subjects online experiments with Amazon Mechanical Turk workers to test attractors in realistic security scenarios. We asked participants to evaluate games on three third-party websites. Unbeknownst to them, we operated the third such website, which appeared to require Microsoft's Silverlight browser plugin. In some cases, the dialog contained a clue that should have made users suspicious about installing. In Experiment 1, we sometimes changed the contents of the publisher field from **Microsoft** to **Miicr0s0ft**. In Experiment 2, we displayed the set of permissions required for the plugin, and some participants were shown egregious permissions being requested.

Our preferred attractors outperformed the control, with statistically significant effects in Experiment 1 and with similar effect sizes (but smaller sample sizes) in Experiment 2. They reduced the proportion of participants choosing the less safe option by up to 50%. Participants who saw our preferred attractors were two to three times more likely to notice the updated message the first time it appeared than those in the control conditions.

6.2 Attractors

An attractor is an interface modification designed to draw attention to a region of the screen. We investigated attractors designed to draw attention to an information field in a decision dialog that is essential to making a good decision. We call this the *salient field*. The attractors we designed are illustrated in Figures 6.2 through 6.7, in which they appear in the context of Experiment 1 (software installation with benign and suspicious publishers).

We designed five *inhibitive attractors*, which prevent a user from making a potentially-hazardous choice until either some period of time has expired or the user performs a required action. These

Figure 6.2: 'Animated Connector' attractor, shown right after the highlighting is activated by hovering the mouse over the install option.

inhibitive attractors appear only when users move their mouse over the button representing the potentially-hazardous choice, which we henceforth refer to as the *triggering option*. The user is never inhibited from closing the dialog or selecting a non-triggering option.

The **Animated Connector** (AC) is a yellow highlight that first appears behind keywords in the triggering option that relates to the salient field. In the installation dialog of Experiment 1, the highlighted keywords in the triggering option text were "this publisher" (Figure 6.2); whereas, in Experiment 2 the highlighted words were "upgraded permissions" (Figure 6.11). Over a period of two seconds, the highlighted region progresses in the direction of the salient field, and then fills the background of the field — hopefully bringing the user's attention with it. Figure 6.2 shows the contents of the decision dialog in Experiment 1 after the animation completed. This attractor is inhibitive when used with a delay that prevents users from proceeding until the animation completes, or when used in combination with another inhibitive attractor.

The **Reveal** attractor (Figure 6.3) first hides the contents of the salient field, then progressively animates it back into place over a period of four seconds. The animation is a progression in which characters are revealed mostly, but not entirely, from left to right. The motion and randomization are intended to help users notice each letter as it appears. Figure 6.3 shows the progressive reveal in mid-flight, and Appendix D.1 provides details on the animation algorithm. The triggering option is disabled until the contents of the salient field have fully appeared and the animation completes.

The **Swipe** attractor (Figure 6.4) requires users to move their mouse over the salient field, from left to right, to enable the triggering option. As the user moves over the letters, they become highlighted.

Because inaccuracies in the user's vertical mouse position are likely to grow as they swipe to the right, the accepted vertical target grows along the x axis. If the user moves her mouse over the triggering option before swiping, a pop-up message appears explaining how to swipe, with an animated cursor illustrating the motion. A green arrow appears beneath the publisher to indicate to

Figure 6.3: 'Progressive reveal' attractor, shown right after being activated by hovering the mouse over the install option. The install option remain disabled until the animation completes.

users that they will need to swipe over the content if they wish to enable the triggering option. The premise behind the *Swipe* attractor is that when users must move the cursor between two points, their attention will be drawn there.

The **Type** attractor (Figure 6.5) requires the user to retype the contents of the salient field. The requirement is not case-sensitive and, for the task of Experiment 1, participants need only type the publisher name and not the domain name. Some websites already ask users to retype their name in order to sign a document. We disabled paste functionality to prevent users from copying and pasting the publisher name without paying attention to it. While we expected this attractor to be quite annoying, its use might be appropriate in situations where the consequences of a mistake are particularly severe. We also intended this attractor to provide a bound on what can be achieved by drawing users' eyes to the salient field, as participants presumably could not type the contents without having read them.

The **Request** attractor (Figure 6.6) uses only a small, secondary pop-up to ask the user to look at the salient field. Our purpose was to establish what an inhibitive attractor can accomplish without animations or input requirements that force users to interact with the salient field. This attractor only required users to click OK to acknowledge the pop-up.

To measure whether the inhibitive feature of attractors is effective, we also sought to include the best 'static' or non-inhibitive attractor from among the many design options that draw attention without inhibiting users' actions. We investigated best practices for drawing attention to physical-world warnings such as road signs and poison labels. The ANSI guidance for warnings recommends high contrast font colors and backgrounds [68, 98], so we created an **ANSI** attractor (Figure 6.7) by using yellow on black text to draw attention to the salient field.

In Figure 6.10 we present all attractors classified according to three features. *Peril-sensitive* attractors are those shown only when the user hovers her mouse over the triggering option. *Inhibitive* attractors are those that prevent the user from picking the triggering option until an action has been

Figure 6.4: 'Swipe' attractor. The install option has been just hovered, which makes the small pop-up appear. In addition, an animation of a cursor moves from left to right along the publisher to show the motion that will be necessary to activate the install option.

completed. *Forced-action* attractors are those in which the action to be completed depends on the user.

6.3 Experimental design

Our experiments share a common ruse in which participants are not told they are participating in a security study, are asked to evaluate third-party websites offering online games, and eventually arrive at a site that triggers a security dialog designed to appear as if it originates from the participant's browser or operating system. This ruse, similar to the one my co-authors and I used in the experiments presented in Chapter 5, is designed to elicit behavior in response to what appears to be a genuine risk to the participant. We used a between-subjects design in which participants saw only one security dialog, as repeated exposures would lead participants to suspect we were studying these dialogs.

6.3.1 Methodology

We recruited participants on Amazon's Mechanical Turk crowdsourcing system. We required participants to connect from an IP address within the U.S., to be at least 18 years old, and to use Chrome, Firefox, or Internet Explorer 9 (for compatibility with our dialog-rendering engine). Since the security dialog was designed to have the look and feel of Microsoft Windows Vista/7, we recruited participants who were using these operating systems. We paid $1.00 to each participant who qualified for and completed our study. The instructions we gave our participants are in Appendix D.2.1.

Figure 6.5: 'Type' attractor. The text box and the small pop-up with instructions appear only if the user hovers the install option.

Tasks

We asked participants to spend two to three minutes evaluating three online games and to report characteristics such as age-appropriateness and the presence of rendering bugs. With each task we presented a form that contained a link to a third-party gaming site and questions about the game. The link text was the URL that the user would be directed to, which led to a third-party domain (see Appendix D.2.2). This illustrated that the participant would be taken to a third-party site outside of our control. To reinforce the impression that the site was outside the control of the researchers, we placed a disclaimer under each link: "By clicking on this link you acknowledge that the website you will be directed to is in no way affiliated with Carnegie Mellon University, and that CMU is in no way responsible for the content of this website." We asked participants to click on the URL, play the corresponding game for two to three minutes, and then come back to the survey to answer questions about the game.

The first two gaming websites we directed participants to evaluate were, in fact, third-party websites over which we had no control. The third, however, was a confederate website that we controlled and that only appeared to be from a third-party: `www.yourgamefactory.net`. In Experiment 1, we displayed a message on the site explaining that "This game requires the latest version of Microsoft Silverlight (v5.1.2). Silverlight is either missing or out of date. Your download will begin in a moment..."[1] After eight seconds of "downloading" our website presented the participant with a dialog, designed to look like an OS-level dialog window, which we rendered within the browser's content region. This dialog asked participants to approve the installation of software. In Experiment 2, the message explained "This game is requesting permission to access a local resource" and participants were asked to grant permissions to a Flash application.

[1]The latest version of MS Silverlight at the time this experiment was performed was 5.1.1.

Figure 6.6: 'Request' attractor. When the user hovers the install option, the small pop-up appears; the user has to click on the small 'OK' button to make it disappear and to enable the install option.

Scenarios

We presented participants assigned to *benign* scenarios with information that would lead them to believe that it was safe to play the game; whereas, participants in *suspicious* scenarios received clues that proceeding might lead to harm.

In Experiment 1, the post-download dialog was an installation dialog in which the salient field contained the name of the software's publisher. We chose an installation dialog because they are familiar to users. They also contain only one field that might provide clues of suspicious behavior: the publisher. Whereas a software publisher can give a program any name it chooses, the publisher's own name must be signed by a certificate authority. Having only one field of salient information to draw users' attention to simplifies experimental design. In the benign scenario the publisher field contained the expected publisher: "Microsoft Corporation (microsoft.com)." The alternative field contents, "Miicr0s0ft Corporation (miicr0s0ft.com)," provided what we hoped would be a suspicious enough clue that installation might be unwise.

In Experiment 2, the post-download dialog was a permission-request dialog, for which the salient field contained the set of permissions required by the game. While Windows and Windows-based browsers do not normally present such dialogs, they are present in mobile operating systems and so we believed a sufficient proportion of participants would perceive the dialog to be legitimate. This decision to grant permission depends both on trusting the provider of the game, which did not vary between conditions, and on understanding the implications of the permissions to be granted, which did vary. The permission field contained "Storage: website cookie" in the benign scenario and "Storage: Access to all files and folders" in the suspicious scenario.

We simulated dialogs in the browser using HTML, CSS, and Javascript. We emulated the look and feel of the Windows Vista/7 interface, including matching the translucent window elements introduced in Windows Vista, supporting dragging of the window (only within the browser content region), and blinking the dialog when the user clicked outside of it (only within the browser content

Figure 6.7: 'ANSI' attractor.

region).

Instrumentation

We instrumented our confederate gaming website to record participants' OS type, browser client name and version, screen size, browser viewport size and zoom level, and the position of the top left corner of the browser's viewport relative to the top left corner of the screen. We also recorded participants mouse movements and clicks within the page content (which included our installation dialog).

Post-Task Survey

The experiment concluded when participants chose an option in the security dialog. In the event that they chose the option that would allow them to continue to play the game, we presented a message that indicated that the game had been taken offline. We then expected them to return to the game evaluation form and answer "no" to the first question, which asked whether they were able to play the game, and to explain why.

After participants completed the evaluation form for the game on our confederate website, we asked them to fill out a post-task survey (see Appendix D.3 for the survey used following the task of Experiment 1, and Appendix D.3 for the exit survey after Experiment 2.) We first asked them whether they had "seen any windows that asked if you wanted to allow software to be installed on your computer?" If they answered yes, we then asked if they had installed the software. In the rare but inevitable instances in which participants reported behavior that did not match our recordings of what they had done, we inspected our records of their mouse movements and clicks to verify that we had assessed the behavior correctly.

We also asked the participant to recall the information from the salient field from a set of options, to determine whether participants who opted to install the software had made an *informed*

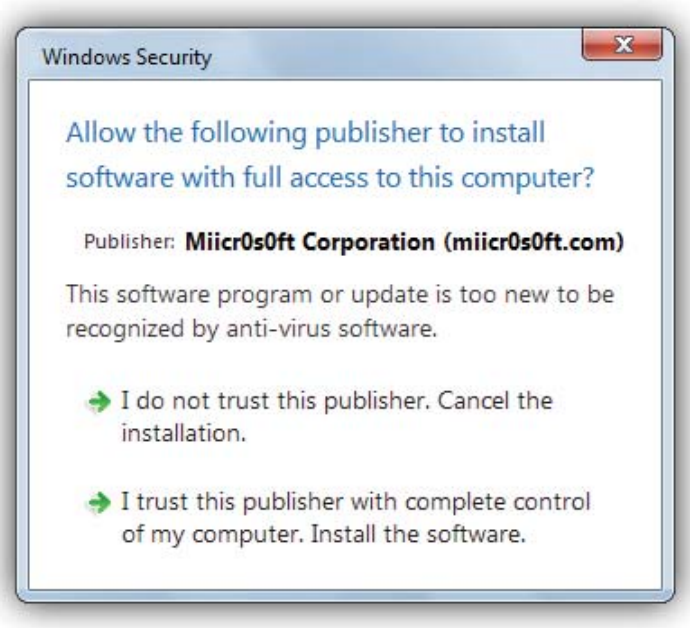


Figure 6.8: 'No antivirus' dialog. This dialog does not include any attractors or animations.

decision, or whether they were *uninformed*. If participants reported seeing clues that should have made them suspicious but installed the software anyway, we asked them why they did so.

Following the questions, we provided a debriefing to dehoax participants and to explain why we believed the use of deception was necessary (see Appendix D.5.) We also asked participants questions designed to elicit reports of any unexpected harm.

Metrics

In *all* of our experiments, each participant represents a single trial and yields a single binary outcome (e.g., installed or didn't install in Experiment 1). From each condition, we tally the total number of participants who fall into these two outcomes to create a binomial proportion. If we were to test a simple hypothesis of the presence of a difference between the binomial proportions of two treatments, we could use a 2×2 test (2 outcomes $\times$ 2 treatments) such as a chi-squared test or Fisher's exact test.

However, the desired effect of a treatment is that it reduces the chance that participants disregard dialogs more in the suspicious case than in the benign case. Thus, we measure and present the *reduction* in binomial proportions for each treatment from the benign to the suspicious scenario. For example, in Experiment 1, we measured the proportion of both benign and suspicious installations, and evaluate attractors by the difference between suspicious installations (which we deem undesirable) and benign installations (which we deem desirable). To determine if an attractor is more effective than the control at reducing the relative proportion of installations from the benign to the suspicious scenario, we require a $2 \times 2 \times 2$ analysis: 2 outcomes $\times$ 2 scenarios (benign or suspicious information) $\times$ 2 treatments (an attractor and the control).

Figure 6.9: 'Short options' dialog. This dialog does not include any attractors or animations.

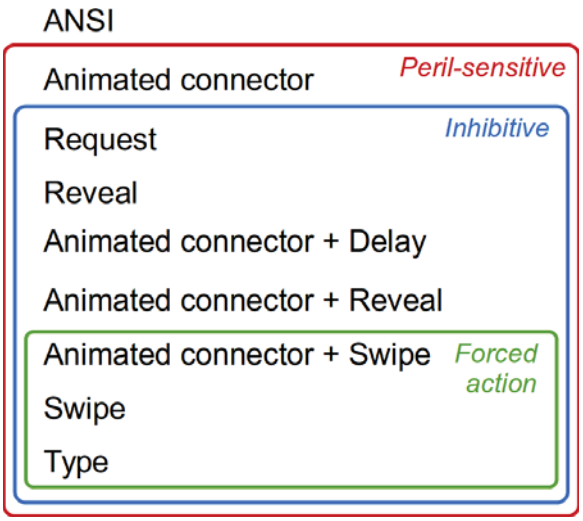


Figure 6.10: Classification of attractors according to their features. *Peril-sensitive attractors* are those shown only when the user hovers her mouse over the triggering option. *Inhibitive attractors* are those that prevent the user from picking the triggering option until an action has been completed. *Forced-action attractors* are those in which the action to be completed depends on the user.

To test whether a treatment had *the desired* effect, we state as a null hypothesis that any change in the proportion of participants who disregarded the dialog from the control to the treatment was *independent* of the scenario (benign or suspicious). In other words, the null hypothesis implies that either (1) the treatment's effect did not differ from that of the control or (2) the effect it did have relative to the control was independent of whether the scenario was benign or suspicious. If there is no change relative to the control, or the change in the proportion of participants was

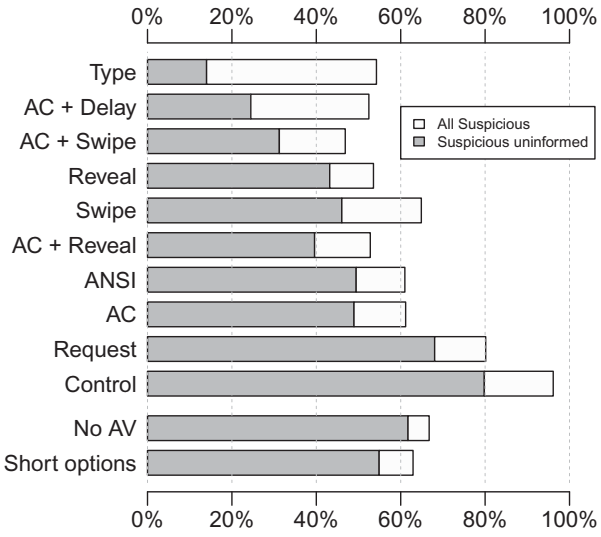
Figure 6.11: Permission dialog used in the benign scenario in Experiment 2. In the suspicious scenario, the requested permissions were for accessing "all files and folder in this computer."

the same regardless of whether the scenario was suspicious, then the treatment did not have the desired effect. The alternate hypothesis is that the treatment had an effect relative to the control and the size of the effect was dependent on the scenario being suspicious—a second-order interaction between treatment and scenario.

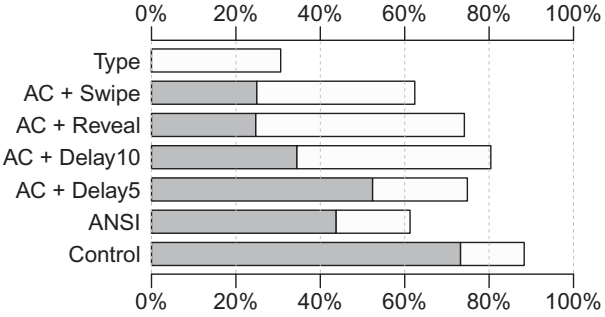
To test a null hypothesis that no second-order interactions are present among two sets of binomial outcomes (a $2 \times 2 \times 2$ table), we use the standard approach of building a log-linear model without second-order interactions and performing a likelihood-ratio test to disprove the hypothesis that the constructed model fits the observed data.

We use this approach in Experiment 1, in which the binomial outcome represents the participant's installation choice (installed *vs.* did not install) and the two independent variables are the scenario (benign *vs.* suspicious) and treatment (attractor *vs.* control or attractor *vs.* attractor). In order to account for the possibility that the increased novelty of certain attractors might cause some participants to want to experiment with the install option in the suspicious case, we also run our analysis a second time using the proportion of participants in the suspicious group who installed and, when asked to identify the publisher, failed to choose "Miicr0s0ft" from the list of five options (see question 6 in Appendix D.3.) In other words, we used the *uninformed* installation rate for the suspicious case.

We use the same approach in Experiment 2, for which the binomial outcome represents whether the participant granted a permission request.



(a) Experiment 1: Suspicious install rate / benign install rate

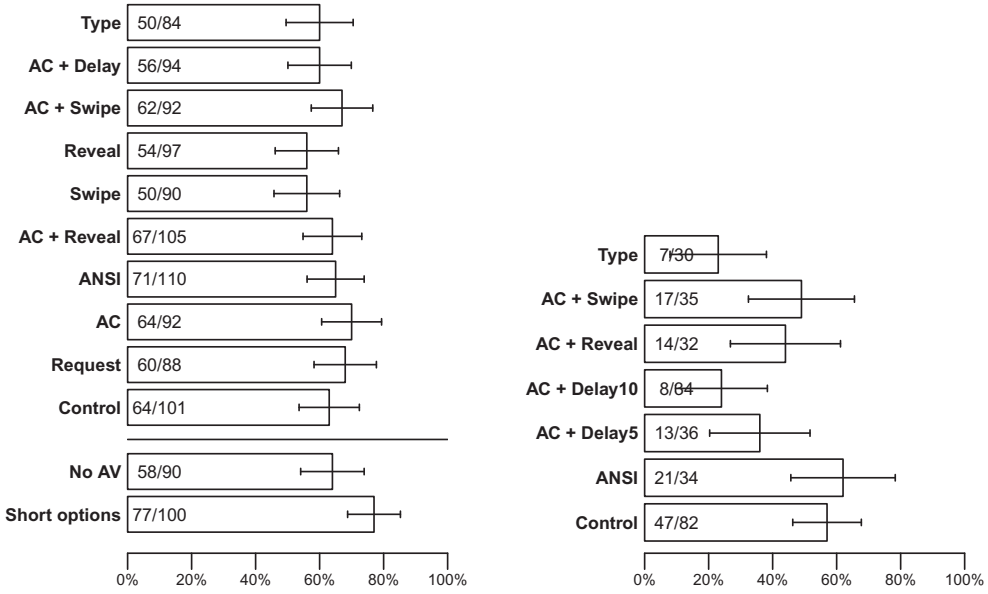


(b) Experiment 2: Suspicious grant rate / benign grant rate

Figure 6.12: In Experiments 1 and 2, an effective attractor should reduce the rate at which participants disregard the dialog in the suspicious scenario, continuing to install software (Exp. 1) or grant permissions (Exp. 2). The white bars represent the rates at which participants disregard the dialogs in the suspicious scenario as a fraction of the rate with which they do so in the benign scenario, whereas the gray bars represent those who were uninformed when they disregarded the dialog — those who were not able to identify the suspicious publisher (Exp. 1) or permission (Exp. 2) when responding to a post-task question. The data from which this graph was generated can be found in Figures 6.13, 6.14 and 6.15.

6.4 Experiment 1: Installing Software

In our first experiment, we presented users with a dialog with two options: installing the software that had been downloaded or canceling the installation. In addition to explicitly choosing the cancel option, users could also avoid installing the software by clicking on the close box (with a red "X") at the top right corner of the dialog or closing their browser tab.



(a) Experiment 1 installation rate.

(b) Experiment 2 permission-granting rate.

Figure 6.13: Performance metrics for Experiments 1 and 2, benign scenario. Left graph (Figure 6.13a) shows the fraction of participants who chose to install the software in the benign condition (the benign install rate). Right graph (Figure 6.13b) shows the fraction of participants who chose to grant permissions in the benign condition (the permission-granting rate). Each bar represents a fraction of the participants in a given condition, and the exact fraction appears within the bar (denominators are the total number of participants in each condition). Whiskers show 95% confidence intervals.

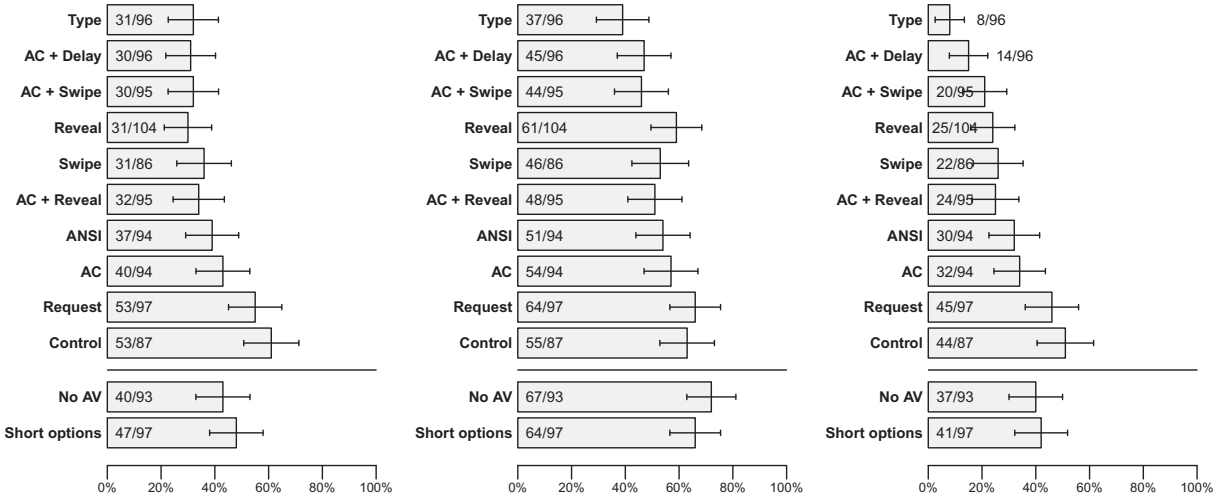
6.4.1 Conditions

We designed 12 treatments, each of which was presented in the context of a benign and a suspicious condition (a total of 24 conditions).

Our control treatment (Figure 6.1) did not use an attractor. The only emphasis given to the publisher field was the use of a bold font for the field's label, which was applied to match the way this field is presented in the User Account Control (UAC) installation dialog in Windows 7. The boldface label appeared in *all* treatments.

We created a treatment for each of the single attractors: *ANSI*, *Animated Connector*, *Reveal*, *Swipe*, *Request*, and *Type* as described in Section 6.2. We created a new treatment, *Animated Connector + Delay*, which disabled the installation option until five seconds after the animation began, effectively turning *Animated Connector (AC)* into an inhibitive attractor. We also included two treatments in which the animated connector was drawn over a period of two seconds and then followed by another inhibitive attractor. In the *Animated Connector + Swipe* condition, it was followed by the *Swipe* attractor; while in the *Animated Connector + Reveal* condition, it was followed by a three-second *Reveal*, resulting in a total delay of five seconds (matching *Animated Connector + Delay*).

Finally, we created two treatments to examine hypotheses *orthogonal* to the efficacy of attrac-



(a) Installation rate.

(b) Publisher-identification error rate.

(c) Suspicious uninformed consent rate.

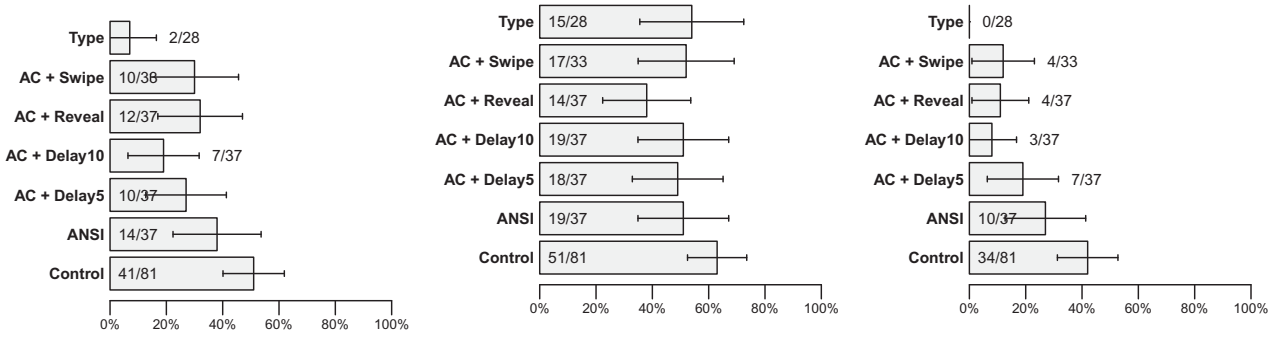
Figure 6.14: Performance metrics per treatment for Experiment 1, suspicious scenario. The graph on the left shows the fraction of participants who chose to install the software, the graph in the center shows the fraction of participants who failed to correctly answer a multiple choice question that asked them to identify the publisher, and the graph on the right shows the fraction who did both. Each bar represents a fraction of the participants in a given condition, and the exact fraction appears within the bar (denominators are the total number of participants in each condition). Whiskers show 95% confidence intervals.

tors. The *Short Options* treatment (Figure 6.9) simplifies the text of the options to "cancel the installation" and "install the software." The advice provided by the dialog, specifically that the installing software gives the publisher "complete control of my computer," is moved from the option to an instruction segment below the publisher name. This treatment explored the hypothesis that users will be more likely to read a dialog with succinctly written options.

The *No Antivirus* (Figure 6.8) treatment explored a hypothesis developed during piloting. We found that those who installed software even after recognizing errors in the publisher name often stated that they felt safe doing so because they had antivirus software installed. This condition was identical to the control except for the following instruction below the publisher: "This software program or update is too new to be recognized by anti-virus software."

6.4.2 Participants

We ran our experiment between August 12, 2012 and September 15, 2012. A total of 4048 eligible Mechanical Turk workers began our study and 2277 encountered the security-decision dialog. We excluded from our results 1771 participants who did not reach the security decision. Our participants were 28.6 years old on average (σ=9.3 years), 54% male, and 75% caucasian. The top two reported occupations were 'student' (27%) and 'unemployed' (17%). 23% reported having knowledge of computer programming. The average completion time was 17 min 22 sec. According to browser user-agent strings, 52% of our participants used Chrome, 36% used Firefox, and



(a) Permission-granting rate.

(b) Permission-identification error rate.

(c) Suspicious uninformed consent rate.

Figure 6.15: Performance metrics per treatment for Experiment 2, suspicious scenario. The graph on the left shows the fraction of participants who chose to grant permissions to the software, the graph in the center shows the fraction of participants who failed to correctly answer a multiple choice question that asked them to identify the permissions being granted, and the graph on the right shows the fraction who did both. Each bar represents a fraction of the participants in a given condition, and the exact fraction appears within the bar (denominators are the total number of participants in each condition). Whiskers show 95% confidence intervals.

12% used Internet Explorer.

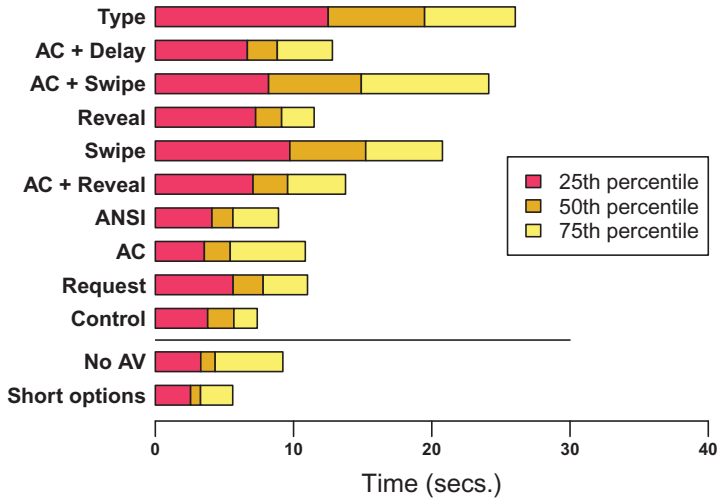
6.4.3 Results

We evaluate attractors by the level with which they reduce installations in the suspicious scenario *relative to* the benign scenario. We do this because reducing installations in the suspicious scenario does not necessarily indicate that an attractor has succeed in capturing participants' attention — it may have simply made it harder to proceed regardless of whether the user should be alarmed by the content of the dialog.

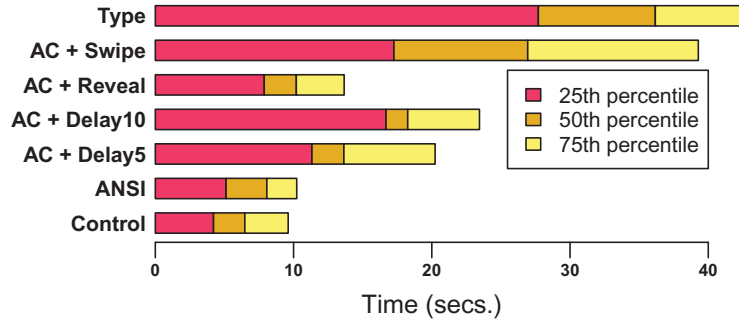
Figure 6.12a presents the reduction in the suspicious-scenario installation rate for each treatment as a fraction of the benign-scenario installation rate. The bars are split into two to represent the uninformed installations (gray bars), which includes participants in the suspicious scenario who failed to select 'Miicr0s0ft' when asked to identify the publisher from a list, from *informed* installations (white bars).

All six inhibitive attractors resulted in greater reductions in the installation rate (from benign to suspicious) than the control. Hypothesis tests, presented in Table 6.1, exceeded the 95% threshold of statistical significance for all but *Swipe*, though *Swipe* exceeded the threshold when joined with an animated connector. The same was true when we revised the test to use only uninformed suspicious installations.

We were surprised at how well *ANSI* performed, as it had not done well during our pilots. Indeed, only *Type* performed significantly better than *ANSI*, and only when excluding informed suspicious installations. However, the results of our first habituation experiment, described in Chapter 7, shed doubt onto the performance of this attractor once users become habituated to seeing it.



(a) Benign-scenario consent delay times for installation dialog (Experiment 1).



(b) Benign-scenario consent delay times for permission dialog (Experiment 2).

Figure 6.16: In Experiments 1 & 2, we show for the benign scenario the 25^{th}, 50^{th}, and 75^{th} percentiles of the time participants lost as a result of having to interact with dialogs that did not convey a cause for suspicion (the benign-scenario consent delay).

The benefits of inhibitive attractors come at a cost, as they delay users who want to disregard a dialog. We present the 25^{th}, 50^{th}, and 75^{th} percentile delays for each treatment in Figure 6.16. As expected, retyping the publisher name (*Type*) takes the most time. The *Swipe* treatment was the second worst offender. However, we hoped that once users learn to recognize the swipe affordance (a green arrow beneath the publisher), and know to swipe before choosing the triggering option, much of the delay introduced by this attractor may disappear, as the results from our first habituation experiment (Chapter 7) suggest.

6.5 Experiment 2: Granting Permissions

To test the robustness of our initial results to other security decisions, we ran a second experiment in which we used attractors in the context of the permission-request dialog in Figure 6.11. We presented this dialog requesting "upgraded permissions" at the same confederate gaming website as in the first experiment, also following the illusory download event. In the benign scenario, we set the contents of the requested-permission field to "Storage: website cookie." In the suspicious scenario we set the field to request "Storage: all files and folders in this computer." The option that triggered an attractor was titled "Yes, run the application with upgraded permissions." When an animated connector was used, the animation began underneath the words "upgraded permissions."

6.5.1 Conditions

We included the following treatments that were the same as Experiment 1: *Control*, *ANSI*, *Animated Connector + Reveal*, *Animated Connector + Swipe*, and *Type*. Given their relatively poor performance, we did not re-test *Request*, *No Antivirus*, and *Short Options*.

In Experiment 1, we were intrigued by how well *Animated Connector + Delay* performed and, in Experiment 2, decided to test whether this relative success was due to the delay introduced. We included two conditions: *Animated Connector + Delay (5 seconds)* and *Animated Connector + Delay (10 seconds)*. The former was the same as in Experiment 1, while the latter doubled the total delay time.

6.5.2 Participants

We ran our experiment between November 05, 2012 and November 15, 2012, again recruiting from Mechanical Turk. We recruited 638 participants, and 573 stayed with the task long enough to encounter the security-decision dialog (65 participants did not reach the security dialog). Our participants were 29.5 years old on average (σ=9.2 years), 49% male, and 75% caucasian. Again, the top two reported occupations were 'student' (22%) and 'unemployed' (14%), and 21% reported having knowledge of computer programming. The average completion time was 17 min 58 sec, and according to browser user-agent strings, 52% of our participants used Chrome, 35% used Firefox, and 13% used Internet Explorer.

6.5.3 Results

Figure 6.12b presents the reduction in the permission-granting rate for each treatment from the suspicious to benign scenario. As with Experiment 1, the bars are split into two in order to represent the uninformed permission grants (the first part of the bar), which includes participants in the suspicious scenario who failed to identify 'All Files and folder on the computer' when asked to identify the requested permission from a list, from *informed* grants (the second part of the bar).

Adding an additional five-second delay to the animating connector did not appear to be valuable, as grants decreased in the benign scenario as much as they did in the suspicious scenario. With the smaller sample size, and slightly better performance by the control group, the improvements over the control were only significant for *Type*, and then only when informed suspicious

permission grants were discarded (See Table 6.2). Again, we were surprised by the relatively good performance of *ANSI*.

The delay time imposed in the benign scenario also mirrored the results of Experiment 1, with *Type* the slowest and *Swipe* the second slowest.

6.6 Limitations

Experiments 1 and 2 were intended to evaluate behavior in a relatively realistic security dialog scenario; however, their validity still is limited. First, we deemed it a success when participants installed Silverlight from "Microsoft," and chose not to install software from "Miicr0s0ft." However some participants may have different definitions of success. For example, some participants presented with the benign scenario might not want to update Silverlight. However, such participants had an equal probability of being assigned to any treatment. Some participants might knowingly install software they deemed suspicious to see what happened. We rely on the fact that participants whose definition of success may have differed from ours had an equal probability of being assigned to any of our treatment groups, thus they are effectively noise in our comparisons.

Some participants might also have detected that the installation dialog was fake, and reflected this insight in their behaviors. We asked participants if they noticed the deception in the post-task survey. Excluding these participants would filter out those who knew the dialog was fake, but would also filter out those participants who had convinced themselves that they were more observant than they actually were. Since we did not see any noticeable differences after filtering out these participants, we chose not to exclude them.

While we restricted participation to participants from IP addresses in the United States, some of our participants might not have been native English speakers and might not have been interacting with their computer in English. If they were not, this would have also made the installation dialog suspicious. Although we found no mention of this issue in the exit surveys, in future studies we would collect the `accept-language` header of the browser and remove participants who are not using a language supported by our study.

These experiments were an investigation of the effectiveness of attractors; we did not aim to create the perfect software-installation or permission-granting dialog. While the ubiquity and simplicity of software-installation dialogs make them an excellent platform for studying attractors, they are becoming less common and less important as operating systems evolve. Operating systems are increasingly incorporating application stores as a recommended means of software installation, as this model offers users more context for making decisions (e.g., feedback from other users). Even when software is downloaded via browsers, operating systems now whitelist popular software and blacklist others, so that users only need to make decisions when installing less popular titles. Even if installation dialogs as we know them become a relic of the past, our findings on the impact of attractors can be applied to those security decisions that users must make in the future.

6.7 Conclusions

We found that inhibitive attractors significantly reduced the likelihood that participants would a) install software despite the presence of clues indicating that the publisher of the software might

not be legitimate, and b) grant dangerously-excessive permissions to an online game.

While inhibitive attractors show promise for directing users' attention to salient features in security dialogs, their use does come at a cost. Even when no risk is present, inhibitive attractors may discourage users from performing useful actions or delay their workflow. Indeed, all inhibitive attractors delayed users' workflow. Fortunately, our habituation experiments (presented in Chapter 7) also showed that the delay incurred by attractors decreases with repeated exposure, especially for the *Swipe* attractor. While the *Swipe* attractor added 3 to 4 seconds of delay even after users learned how to use it, some delay is unavoidable if an attractor accomplishes its purpose: forcing users to read the portion of a dialog that might allow them to discover security risks they had not expected.

Treatment names		Benign		Suspicious			Susp. Uninf.		
A	B	R_b^A	R_b^B	R_s^A	R_s^B	$P\left[\frac{R_s^A}{R_b^A} = \frac{R_s^B}{R_b^B}\right]$	R_u^A	R_u^B	$P\left[\frac{R_u^A}{R_b^A} = \frac{R_u^B}{R_b^B}\right]$
Are inhibiting attractors better than Control?									
Control	AC	64—37	64—28	53—34	40—54	$= 0.0171$	44—43	32—62	$= 0.0254$
Control	Swipe	64—37	50—40	53—34	31—55	$= 0.1078$	44—43	22—64	$= 0.0813$
Control	Reveal	64—37	54—43	53—34	31—73	$= 0.0199$	44—43	25—79	$= 0.0453$
Control	AC + Swipe	64—37	62—30	53—34	30—65	$= 0.0012$	44—43	20—75	$= 0.0006$
Control	AC + Reveal	64—37	67—38	53—34	32—63	$= 0.0068$	44—43	24—71	$= 0.0085$
Control	Type	64—37	50—34	53—34	31—65	$= 0.0179$	44—43	8—88	< 0.0001
Are inhibiting attractors better than ANSI?									
ANSI	AC	71—39	64—28	37—57	40—54	$= 0.8214$	30—64	32—62	$= 0.7617$
ANSI	Swipe	71—39	50—40	37—57	31—55	$= 0.5798$	30—64	22—64	$= 0.8815$
ANSI	Reveal	71—39	54—43	37—57	31—73	$= 0.8983$	30—64	25—79	$= 0.9598$
ANSI	AC + Swipe	71—39	62—30	37—57	30—65	$= 0.2726$	30—64	20—75	$= 0.1222$
ANSI	AC + Reveal	71—39	67—38	37—57	32—63	$= 0.6077$	30—64	24—71	$= 0.4933$
ANSI	Type	71—39	50—34	37—57	31—65	$= 0.8238$	30—64	8—88	$= 0.0047$
Are other inhibiting attractors better than Request?									
Request	Swipe	60—28	50—40	53—44	31—55	$= 0.6125$	45—52	22—64	$= 0.3895$
Request	Reveal	60—28	54—43	53—44	31—73	$= 0.2331$	45—52	25—79	$= 0.2767$
Request	AC + Swipe	60—28	62—30	53—44	30—65	$= 0.0348$	45—52	20—75	$= 0.0115$
Request	AC + Reveal	60—28	67—38	53—44	32—63	$= 0.1173$	45—52	24—71	$= 0.0875$
Request	Type	60—28	50—34	53—44	31—65	$= 0.208$	45—52	8—88	$= 0.0002$
Does Reveal add value above delay?									
AC + Delay	AC + Reveal	56—38	67—38	30—66	32—63	$= 0.8725$	14—82	24—71	$= 0.2858$
Is the Swipe or Reveal attractor better?									
Swipe	Reveal	50—40	54—43	31—55	31—73	$= 0.5013$	22—64	25—79	$= 0.8453$
AC + Swipe	AC + Reveal	62—30	67—38	30—65	32—63	$= 0.5553$	20—75	24—71	$= 0.3867$
Does AC aid other attractors?									
Swipe	AC + Swipe	50—40	62—30	31—55	30—65	$= 0.1097$	22—64	20—75	$= 0.1052$
Reveal	AC + Reveal	54—43	67—38	31—73	32—63	$= 0.7025$	25—79	24—71	$= 0.532$
Does adding another attractor help AC?									
AC	AC + Swipe	64—28	62—30	40—54	30—65	$= 0.3965$	32—62	20—75	$= 0.2226$
AC	AC + Reveal	64—28	67—38	40—54	32—63	$= 0.7833$	32—62	24—71	$= 0.7114$
Did Type outperform composite attractors?									
Type	AC + Swipe	50—34	62—30	31—65	30—65	$= 0.3983$	8—88	20—75	$= 0.1716$
Type	AC + Reveal	50—34	67—38	31—65	32—63	$= 0.7833$	8—88	24—71	$= 0.0288$
Did orthogonal treatments differ from Control?									
Control	Short options	64—37	77—23	53—34	47—50	$= 0.0068$	44—43	41—56	$= 0.0208$
Control	No AV	64—37	58—32	53—34	40—53	$= 0.0705$	44—43	37—56	$= 0.2559$

Table 6.1: In Experiment 1, the installation ratio R is the fraction of participants who chose to install the software over those who did not. The superscript is the treatment (A or B) and the subscript is the scenario (benign, suspicious, or uninformed suspicious). The odds ratio is the suspicious-scenario installation ratio over the benign ratio (a ratio of ratios, R_s/R_b). To determine if one treatment did a better job of reducing installations than another treatment, we attempt to disprove the null hypothesis that both treatments' odds ratios are equal (see Section 6.3.1).

Treatment names		Benign		Suspicious			Susp. Uninf.		
A	B	R_b^A	R_b^B	R_s^A	R_s^B	$P\left[\frac{R_s^A}{R_b^A} = \frac{R_s^B}{R_b^B}\right]$	R_u^A	R_u^B	$P\left[\frac{R_u^A}{R_b^A} = \frac{R_u^B}{R_b^B}\right]$
Are tested attractors better than Control?									
Control	ANSI	47—35	21—13	41—40	14—23	$= 0.2225$	34—47	10—27	$= 0.1513$
Control	AC + Delay5	47—35	13—23	41—40	10—27	$= 0.7982$	34—47	7—30	$= 0.6716$
Control	AC + Delay10	47—35	8—26	41—40	7—30	$= 0.9921$	34—47	3—34	$= 0.4153$
Control	AC + Reveal	47—35	14—18	41—40	12—25	$= 0.719$	34—47	4—33	$= 0.0703$
Control	AC + Swipe	47—35	17—18	41—40	10—23	$= 0.3954$	34—47	4—29	$= 0.0536$
Control	Type	47—35	7—23	41—40	2—26	$= 0.1996$	34—47	0—28	$= 0.0144$
Is AC + Delay10 better than AC + Delay5?									
AC + Delay5	AC + Delay10	13—23	8—26	10—27	7—30	$= 0.8501$	7—30	3—34	$= 0.686$
Does Reveal add value over Delay?									
AC + Delay5	AC + Reveal	13—23	14—18	10—27	12—25	$= 0.933$	7—30	4—33	$= 0.2405$
AC + Delay10	AC + Reveal	8—26	14—18	7—30	12—25	$= 0.7886$	3—34	4—33	$= 0.5305$

Table 6.2: In Experiment 2, the permission-granting ratio R is the fraction of participants who chose to grant permissions to those who did not. The superscript is the treatment (A or B) and the subscript is the scenario (benign, suspicious, or uninformed suspicious). The odds ratio is the suspicious-scenario permission-granting ratio over that of the benign ratio (a ratio of ratios, R_s/R_b). To determine if one treatment did a better job of reducing installations than another treatment, we attempt to disprove the null hypothesis that both treatments' odds ratios are equal (see Section 6.3.1).

Chapter 7

Testing resilience of attractors to habituation

7.1 Introduction

Habituation is frequently blamed for sapping users' attention to security dialogs (see Section 2.1.5 for a discussion on the topic). Having shown previously that attractors were effective at driving participants' attention to a salient field in a dialog, we aim now at testing whether the same attractors are resilient to repeated exposure. In this chapter, we report on two experiments designed to test how resilient attractors are to habituation.

As discussed in Section 2.1.5, we understand habituation as a decrease in attentional response due to repeated exposure to a stimulus [50]. In our case, the stimulus we want computer users to not decrease attention to is the salient field in a security dialog. There are two situations in which this might be problematic, the first one being more concerning than the second. First, if a user evaluates the salient field in a dialog and picks the risky option $n-1$ times, she might get used to pick that option and fail to evaluate the salient field the n^{th} time, when the scenario is risky and the right choice is to not pick the unsafe option. Second, if in a similar situation a user gets used to pick the safe option, she might fail to pick the unsafe option the n^{th} occasion, when the scenario is safe and the right choice is to pick the unsafe option.

We performed two experiments to test whether participants would still pay attention to the salient field in a dialog implementing attractors, after having interacted a large number of times with such a dialog. In Experiment 1, we attempted to habituate participants to dialogs that they knew were part of the experiment. We used attractors to highlight a field that was of no value during habituation trials and contained critical information after the habituation period. Participants exposed to inhibitive attractors were two to three times more likely to make an informed decision than those in the control condition.

Lacking low-habituation conditions, in Experiment 1 we could not prove that attractors actually prevent habituation (as opposed to simply improving performance in both low- and high-habituation conditions). To test the hypothesis that attractors actually prevent habituation, we replicated Experiment 1 with a larger pool of participants adding low-habituation conditions. We

This chapter is a partial reproduction of a paper co-authored with Lorrie Cranor, Julie Downs, Saranga Komanduri, Rob Reeder, Stuart Schechter, and Manya Sleeper [14]; it includes material previously unpublished.

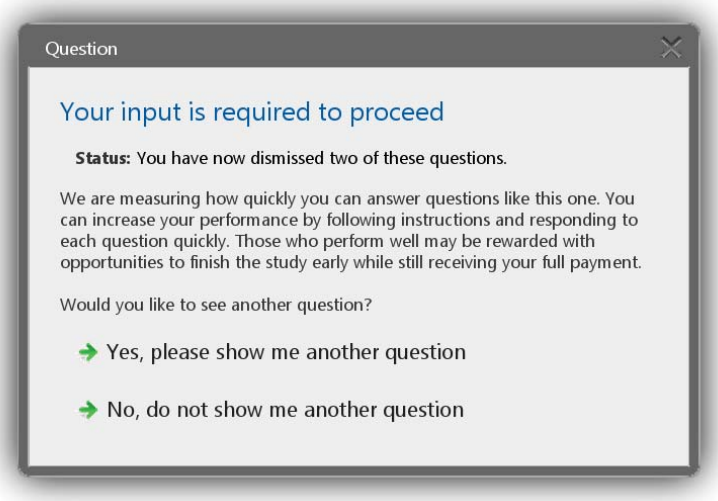


Figure 7.1: Dialog used for Experiment 1. Inhibitive attractors triggered the first (yes) option. When an animated connector was used, it would begin by highlighting the word 'question' in the triggering option.

obtained a stronger finding; whereas habituation can reduce the fraction of participants who pay attention to key information by more than a factor of three when attractors aren't used (the control condition), some attractors eliminate the effects of the habituation period entirely.

7.2 Experiment 1: high-habituation conditions

Having tested attractors on users who had not seen them before, we next tried replicate the effect of habituation to attractors that would result from repeated exposure. Since participants would not have seen our attractors before, we would need to habituate them as part of the experiment. This meant starting over with a new experimental design in which attractors would not be used in a security context — repeated exposures to a dialog would make it effectively impossible to keep participants from figuring out that the dialog was the focus of the study.

7.2.1 Experimental design

We created a task in which participants would first be repeatedly exposed to a dialog during a *habituation period*. During this period, the salient field would not contain information relevant to making a correct decision. After a certain number of habituation exposures, we would inject information essential to the decision into the salient field, testing to see if participants would notice it and make the intended choice. As in previous experiments, we used a between-subjects design in which each participant would yield a single data point. The experiment was approved by Carnegie Mellon University Institutional Review Board.

We recruited workers on Amazon's Mechanical Turk to perform a work task. We instructed them that they would spend five minutes on the task, and that they would be asked to spend the time

answering a dialog as many times as they could. The solicitation text is contained in Appendix E.1, and the instructions given to participants at the beginning of the task are in Appendix E.2.

During a *habituation period*, we displayed the dialog shown in Figure 7.1, for which the contents of the status field alternated between two messages: "You have now dismissed *n* of these questions" and "*n* questions have been dismissed so far," where *n* was the number of dialogs the user had already dismissed, expressed in words. The "no" option and close box were both disabled. Attractors directed attention to the status field, but the number of dialogs dismissed so far was not relevant to the users' decision—the only available option was "yes". We did not inform participants that the task would change during the five minutes we had asked them to perform it.

The habituation period was followed by the test period during which we presented the same dialog, but with the "no" option enabled and the contents of the status field replaced with the following instruction: "Press the No option below to finish this study early." Participants who read and understood the text in the status field discovered that they should stop choosing the "yes" option and instead choose "no." During the test period, the dialog with the updated status field was shown repeatedly until the participant either selected the "no" option or completed their five-minutes commitment. Half way through the test period the instruction was displayed in all capital letters.

To prevent participants from shirking, we excluded those who were inactive for 30 seconds or more. Participants were warned if inactive for 15 seconds. We paid $0.50 to all participants who completed the experiment.

Once participants clicked "no" or the test period expired, we presented them with an exit survey (see Appendix E.3). We asked participants to recall the contents of the status field, instructing those with no recollection to type "None." We used this and other follow-up questions to understand whether participants who never clicked on the "no" option had done so because they had not seen the instruction in the status field or for other reasons, such as misinterpreting the message.

7.2.2 Metrics and conditions

The only metric we used in Experiment 1 is the *immediate detection rate*: the proportion of users who click the "No" option on the first trial of the test period (the first time it appeared). Higher immediate detection rates are better. We understand this metric to be an inverse proxy to habituation: higher levels of immediate detection rate are related to lower levels of habituation.

All statistical testing was done using two-way Fisher's exact test with a significance level of $\alpha = 0.05$ and correcting for multiple tests with the Holm-Bonferroni method.

We tested using attractors from the first experiment: *Swipe*, *Type*, *Animated Connector + Swipe*, *Animated Connector + Delay*, and *Animated Connector + Reveal*. We omitted *Request* given its relative inefficacy in earlier experiments, and we excluded the treatment in which the animated connector was not used with an inhibitive delay.

We expected immediate detection rate to decrease both as a function of the number of times the participant saw a dialog and how long the participant saw the dialog. For most conditions, the habituation period ended when a dialog was dismissed after 150 seconds had passed, which was half way through the five minutes participants were told they would be spending on the task. However, the *Control* dialog and *ANSI* attractor can be dismissed much more quickly than inhibitive attractors. While participants shown inhibitive attractors in our pilots received roughly 22 exposures during the habituation period, participants in the *Control* and *ANSI* received many

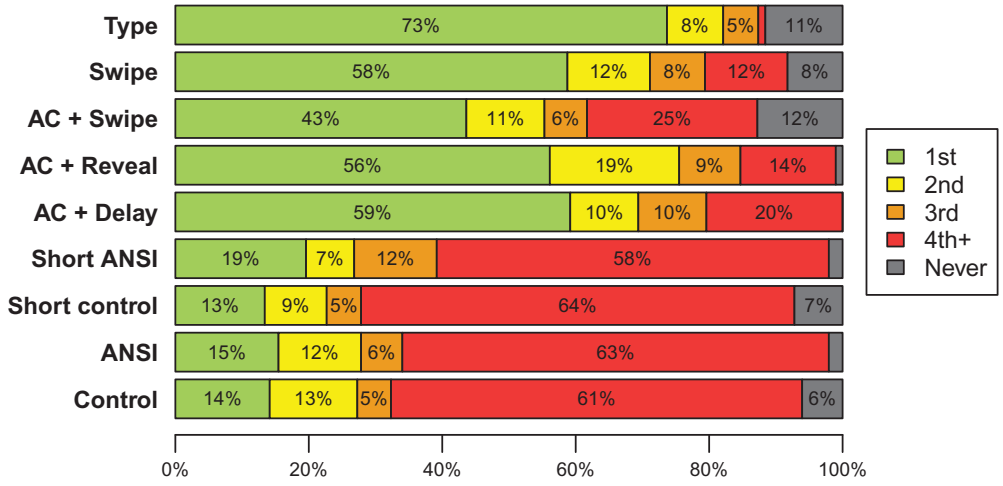


Figure 7.2: Immediate detection rate in Experiment 1: the proportion of participants in each condition who clicked on the 'No' option in response to the first dialog instructed them to do so.

	AC + Delay	*AC + Reveal*	*AC + Swipe*	*Swipe*	*Type*
Control	$p < 0.0001$	$p < 0.0001$	$p < 0.0001$	$p < 0.0001$	$p < 0.0001$
ANSI	$p < 0.0001$	$p < 0.0001$	$p < 0.0001$	$p < 0.0001$	$p < 0.0001$
Short control	$p < 0.0001$	$p < 0.0001$	$p < 0.0001$	$p < 0.0001$	$p < 0.0001$
Short ANSI	$p < 0.0001$	$p < 0.0001$	$p = 0.0005$	$p < 0.0001$	$p < 0.0001$

	AC + Delay	*AC + Reveal*	*AC + Swipe*	*Swipe*
Type	$p = 0.0666$	$p = 0.0468$	$p = 0.0001$	$p = 0.0666$

Table 7.1: Hypotheses comparing relative performance of attractors in Experiment 1. In the top table, each row represents an hypothesis and contains exactly 5 comparisons. The bottom table contains the results of our last hypothesis, comparing Type with the rest of the attractors. All p-values were corrected within each hypothesis using the Holm-Bonferroni method.

more exposures, thus potentially receiving a stronger habituation effect. We, therefore, tested two sets of conditions for *Control* and *ANSI* dialogs, one with the original 150 second habituation period and a pair of short treatments (*Short Control* and *Short ANSI*) that terminated the habituation period after 22 exposures.

7.2.3 Results

We ran this experiment from February 07, 2013 until February 27, 2013. We recruited a total of of 878 participants to the task and 872 finished.

Participants were 30.8 years old on average (σ=11.7 years), 60% male, 77% caucasian, and again the top two reported occupations were "student" (21%) and "unemployed" (16%). According to user agent strings, 50% of participants used Chrome, 40% used Firefox, 6% used Internet

	Median of habit. trials	Median time to complete hab. trials (secs.)						Total participants
		1^{st}	2^{nd}	$25^{th}\%$	$50^{th}\%$	$75^{th}\%$	Last	
Control	54	10.48	6.06	1.36	1.03	0.98	1.05	99
ANSI	50	9.73	6.99	1.3	1.04	1.1	0.98	97
Short control	22	10.66	5.57	1.55	1.34	1.1	1.12	97
Short ANSI	22	11.25	5.39	1.46	1.22	1.24	1.12	97
AC + Delay	15	14.81	11.11	9.21	7.05	6.96	7.46	98
AC + Reveal	15	13.53	10.34	7.79	6.99	7.3	7.38	98
AC + Swipe	18	29.68	8.76	5.38	4.36	4.25	4.49	94
Swipe	17	37.04	10.53	5.68	4.73	4.3	5.14	97
Type	6	57.88	18.21	19.36	16.12	15.65	15.85	95

Table 7.2: In Experiment 1, median number of habituation trials, per condition, and median dialog response times, per condition. With the exception of Short Control and Short ANSI, all conditions are time-based, and thus have a variable number of habituation trials. The second column from right to left shows the last habituation trial before the first test trial (containing the 'No' message.)

Explorer and 4% used Safari. Finally 75% used either MS Windows Vista, 7 or 8, 13% used Mac OS, and 10% used Windows XP, again as reported by their browser user agent strings.

Our results, illustrated in Figure 7.2, show that all five inhibitive attractors had a significantly higher immediate detection rates than the control: between 44% for *AC + Swipe*, and 74% for *Type*, as opposed to the non-inhibitive treatments which reached a maximum of 20% for *Short ANSI*.

Our hypotheses tested whether our inhibitive attractors (*Animated Connector + Delay*, *Animated Connector + Reveal*, *Animated Connector + Swipe*, *Swipe*, and *Type*) displayed a higher immediate detection rate than any of *Control*, *ANSI*, *Short Control* and *Short ANSI*; that is, whether a higher proportion of participants exposed to one of the attractors noticed the "No" message the first time it was shown.

Participants exposed to the inhibitive attractors were significantly more likely to notice the "No" the first time it was shown ($p = .0005$ for the comparison between *Short ANSI* and *Animated Connector + Swipe*, and $p < .0001$ for all other comparisons, for details see Table 7.1). Thus, tested attractors performed well under these conditions of extreme habituation.

As can be observed in Figure 7.2, the *Animated Connector + Swipe* treatment had a lower immediate detection rate (44%) than the rest of the inhibitive attractors. We believe that the conjunction of both the highlighting of the *Animated Connector* and the green arrow behind the text in the status field may have decreased the legibility of the status field for some of our participants in that condition.

As Table 7.2 shows, median times also showed a sharp decrease as the habituation period progresses, regardless of both treatment group and number of habituation trials. This provides evidence that participants quickly learned how to perform the task, and accordingly decreased their response time to dialogs.

We had posited that users would quickly learn to reduce the time they needed to spend responding to the *Swipe* attractor, as an affordance allows them to recognize the presence of the attractor and forgo the time-consuming training message and animation. Whereas the median time to complete the first *Swipe* attractor was 37 seconds, training reduced the time to under five seconds.

7.2.4 Limitations

To habituate participants to attractors they would not have seen before, we needed to abandon the context of a realistic security scenario. Users may behave differently in security situations than they did in responding to these dialogs. We attempted to create a high level of habituation to determine the limits of habituation impact on attractors. However, the levels of habituation in the experiment may not reflect those found in the real world.

Differences in the number of habituation exposures may have led to inconsistent levels of habituation between different treatment groups. However, an analysis of the decreases in per-dialog time in the habituation period showed that the habituation effect was approaching its limits when the test trials began.

Participants who found inhibitive attractors annoying may have had a greater incentive to try clicking 'no' upon noticing that the option was no longer available. This could cause the impact of inhibitive attractors to be overstated.

Finally, we also could not guarantee that participants who saw the message encouraging them to click 'no' would always do so. One participant reported continuing "because the task was fun and I wanted to see how many I could do in the time given."

7.3 Experiment 2: low-habituation conditions

In the first experiment we did not test low-habituation conditions. Thus, we could not conclude that attractors reduced the impact of habituation. It was possible that the margin of superiority of attractors over the control would be present regardless of both before and after habituation had reduced attention in both the attractor and control conditions. We replicated Experiment 1 adding low-habituation conditions to complement the already tested high-habituation conditions. This yielded three new findings:

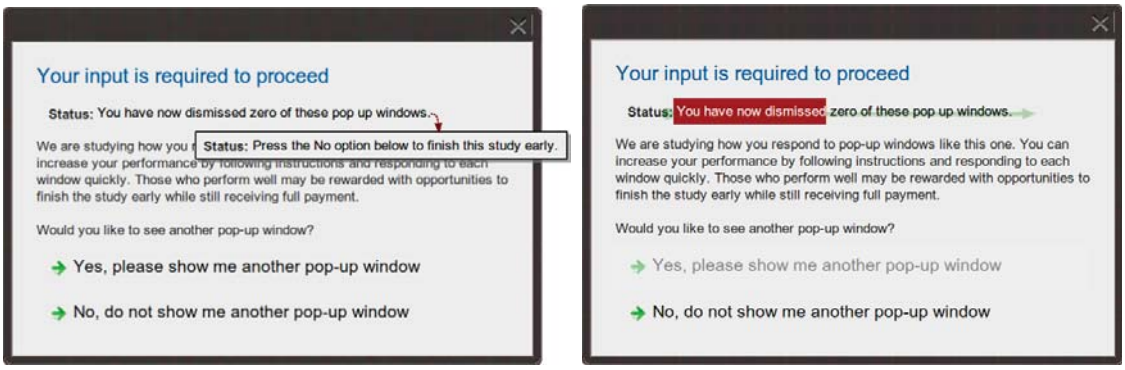
Habituation reduces attention in the experimental task. For the control dialog, increasing habituation decreased the proportion of participants who would choose the *No* option by more than a factor of three.

Some attractors failed to prevent declines in attention. Increasing habituation impacted the performance of participants shown attractors that delayed the activation of the *Yes* option to display an animation, though these attractors still outperformed the control in high-habituation conditions.

Some attractors yielded no declines in attention. Habituation period had no measurable impact on the performance of two attractors that were designed to force users to pay attention to the field in which the instruction to choose *No* appeared.

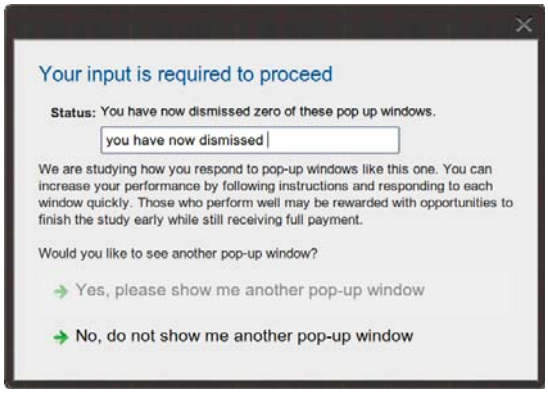
7.3.1 Experimental design

Except for the differences described below, our experimental design was identical to Experiment 1 design. We recruited participants from Amazon's Mechanical Turk, asking workers to perform a task in which they would respond to as many dialogs as possible for a fixed time. Instead of

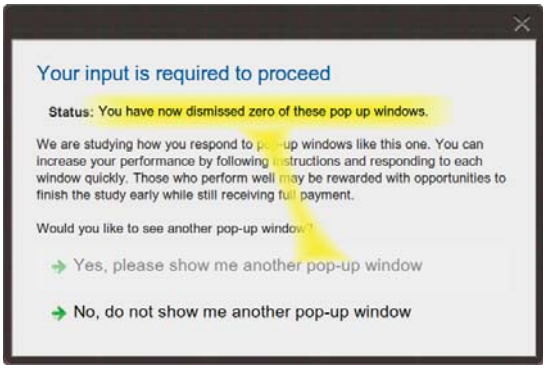


(a) Control dialog

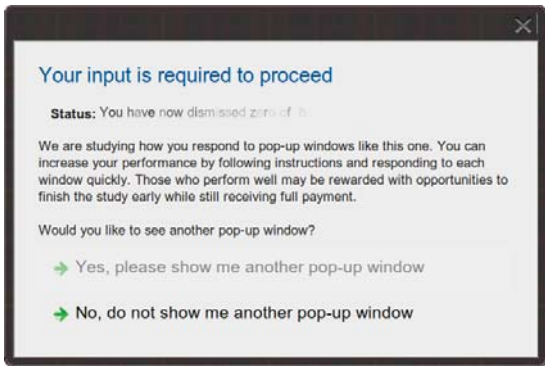
(b) Swipe attractor



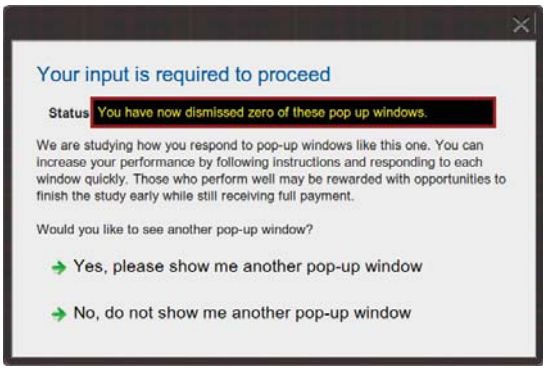
(c) Type attractor



(d) Animated Connector + Delay attractor



(e) Reveal attractor



(f) ANSI attractor

Figure 7.3: Dialogs used in Experiment 2. After the habituation period, the message in the 'Status' field (irrelevant for participants' task) was replaced by a message claiming participants' attention (Figure 7.3a, overlay box).

instructing participants to spend *five* minutes on the task, we asked for *ten* minutes to accommodate longer habituation.

During a *habituation period*, we displayed the dialog in Figure 7.3a. This dialog differs from the one used in Experiment 1 (Figure 7.1) in that we used "pop up windows" in place of "ques-

tions" to describe the dialogs that participants were asked to dismiss. During the habituation period, the information field, labeled "Status" alternated between the message "You have now dismissed *n* of these pop up windows" and "*n* pop up windows have been dismissed so far," where n was written in words. During the habituation period, we disabled the *No* option so participants could only choose *Yes*. We displayed each dialog at random coordinates in a participant's browser. If we detected 15 seconds of inactivity we warned participants that we would exclude those who were inactive for 30 seconds or more.

We followed the habituation period with a *test period*, during which we presented the same dialog, but with the *No* option enabled and the contents of the status field replaced with: "Press the No option below to finish this study early." Participants who read and understood the text in the status field discovered that they should stop choosing *Yes* and instead choose *No*. We terminated the period when the participant chose *No* or ten minutes were up. In contrast to Experiment 1, we did not change the instructions to all capital letters in the middle of the *test period*.

Again, we did not inform participants that the task would change during the ten minutes. After the task, we presented an exit survey in which we asked participants to recall the contents of the status field, instructing those with no recollection to type *None*. Instead of $0.50, we paid $1.00 to participants who completed the experiment.

For the control treatment and each of the five attractors (six treatments), we created conditions for each of four habituation periods, resulting in 24 (6×4) total conditions. We defined the duration of three habituation periods by the number of habituation dialogs the participant would be exposed to (1, 3, and 20 exposures), lasting for as long as it took participants to complete these dialogs. We defined the duration of the fourth habituation period in units of time (150 seconds, plus the additional time required to complete the habituation dialog present at the moment the 150-second period expired), with the number of dialogs varying between participants. We did not create a zero-exposure habituation period because participants would have been entirely unfamiliar with the dialog and attractors. Each participant was assigned to one condition.

For each treatment, we examine the *habituation odds ratio*, the reduction in the proportion of participants who chose *No* from the low-habituation to the high-habituation condition. This yields a $2 \times 2 \times 2$ contingency table: 2 treatments (attractor vs. control) $\times$ 2 habituation conditions (low vs. high-habituation) $\times$ 2 outcomes (the *Yes* or *No* option). To test the null hypothesis that habituation caused the same reduction in the proportion of participants who chose *No* regardless of treatment, we used a log-linear model without second-order interactions and a likelihood-ratio test to compare this model to the observed data. If the observed data deviates significantly from the expected model, it indicates that the treatment might have an effect on the habituation odds ratio.

7.3.2 Results

We recruited 3,071 participants and 2,567 finished. Participants were 29.4 years old on average (σ=10.1 years), 55% male, 77% caucasian, and the top two reported occupations were "student" (25%) and "unemployed" (15%). According to user agent strings, 60% of participants used Chrome, 37% used Firefox, and 3% used Internet Explorer. Finally 73% used either MS Windows Vista, 7 or 8, 15% used Mac OS, and 9% used Windows XP, again as reported by their browser user agent strings.

	Fixed exposure count							Fixed exposure time		
	1 exposure		3 exposures		20 exposures			150 seconds		
	med. time	R_{1e} (*No*·*Yes*)	med. time	R_{3e} (*No*·*Yes*)	med. time	R_{20e} (*No*·*Yes*)	$P\left[\frac{R_{1e}}{R_{20e}} = \frac{R^c_{1e}}{R^c_{20e}}\right]$	med. exp.	R_{150s} (*No*·*Yes*)	$P\left[\frac{R_{1e}}{R_{150s}} = \frac{R^c_{1e}}{R^c_{150s}}\right]$
Control	10 sec	50·56	3.4 sec	43·64	1.2 sec	24·90	—	192	7·99	—
ANSI	10.9 sec	57·55	3.9 sec	49·58	1 sec	15·95	$= 0.1333$	198	13·94	$= 0.3466$
AC + Delay	15.7 sec	89·18	9.8 sec	86·22	6.8 sec	65·43	$= 0.9578$	50	47·60	$= 0.1933$
Reveal	14.2 sec	84·25	8.4 sec	81·22	7 sec	57·47	$= 0.6565$	48	59·47	$= 0.0021$
Swipe	39 sec	61·45	6.9 sec	56·48	3.9 sec	59·48	$= 0.0062$	76.5	65·45	< 0.0001
Type	57.4 sec	79·33	16.6 sec	79·25	12.9 sec	86·13	< 0.0001	24	90·14	< 0.0001

Table 7.3: Median exposure times and habituation odds ratios. The compliance ratio R is the fraction of participants who complied with the instruction to choose the *No* option in the first test trial over those who did not. Control group ratios are written R^c. The habituation odds ratio is the low-habituation compliance ratio over the high-habituation compliance ratio. To determine whether habituation had a greater or lesser effect in a treatment than the control, we attempt to disprove the null hypothesis that their odds ratios are equal.

For each participant, we consider the outcome a success if the participant chose the *No* option in response to the first dialog (test trial) in which they were instructed to do so. The *compliance rate* is the fraction of participants in each condition who complied. We used a binomial outcome representing the result of the first test trial (complied *vs.* did not comply) with the length of the habituation period and the treatment (*attractor* or *control*) as independent variables. We present our results in Table 7.3 and graph the compliance rate as a function of the number of habituation exposures (log scale) in Figure 7.4.

For our *Control* dialog (no attractor), the compliance rate starts low and declines steeply with more habituation exposures. The compliance rates of participants who saw the *ANSI* treatment were not significantly better, and were actually worse (though not significantly so) for the 20-exposure conditions.

The two attractors that impose a delay but do not force the user to interact with the salient field, *Animated Connector + Delay* and *Reveal*, did best in low-habituation conditions, but saw declines in compliance with slopes similar to those seen for the control. Neither was better than control for compliance as habituation increased from one to twenty exposures.

We were surprised to see the compliance rate for *Type* grow with the log of the number of exposures. As habituation increased from one to twenty exposures, the compliance reduction for *Type* (for which compliance increased) versus that of the control was statistically significant, with $p < 0.0001$. A possible explanation is that participants' motivation to comply with the instruction, and to end the experiment early, may have increased as they grew tired of the task. This would represent a countervailing force that overpowers the minimal impact of decreased attention for this attractor. This force might be larger if users are annoyed by an attractor.

The *Swipe* attractor was second most resistant to habituation. As habituation increased from one to twenty exposures, the compliance reduction for *Swipe* was significantly smaller than the *Control* ($p < 0.0062$). However, the lower reduction in compliance relative to *Animated Connector + Delay* and *Reveal* (which were not statistically significantly better than the control) might be due to its inferior performance in the single-habituation case — its performance had less room to fall. For *Swipe*, however, users also quickly became efficient at interacting with the dialog (Fig-

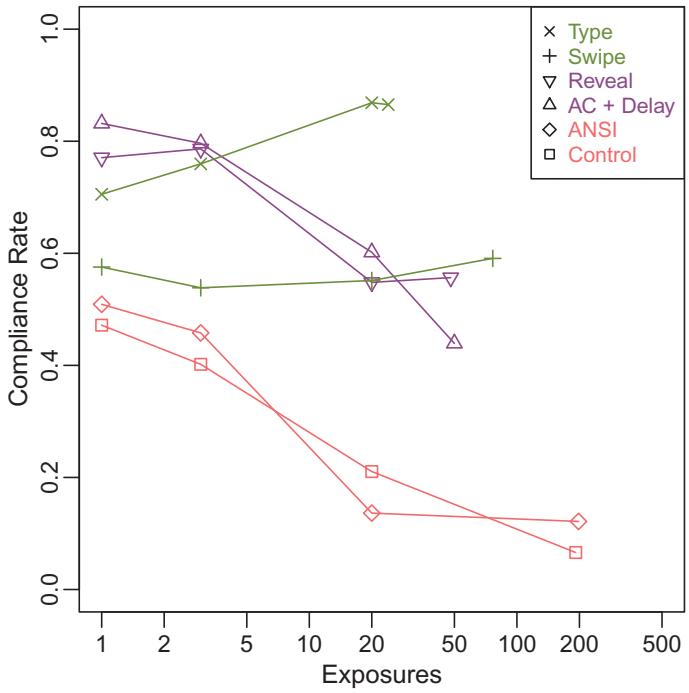


Figure 7.4: Participants' compliance with instruction to click *No* in response to the first dialog in which they were asked to do so. Compliance rate is number of participants who chose *No* over total participants in that condition. The level of habituation is measured by number of exposures to the habituation dialog. The exposures varied for the fixed-time-period (150 s) conditions, so we use median number of dialogs dismissed to plot this point. Downward slopes represent a reduced compliance. The graph was generated from data in Table 7.3. *Control* and *ANSI* do not significantly differ. *Reveal* and *Animated Connector + Delay* are similar, but with an overall better compliance rate; however, neither present significant improvement in compliance rate reductions. Both *Type* and *Swipe* present steady or upward slopes and show significantly better compliance rates as compared to the control.

ure 7.5). After 20 exposures, participants were nearly twice as efficient in interacting with it as they were for delay-based attractors. Nearly 75% of participants using the *Swipe* treatment were able to dismiss the 20th habituation dialog within five seconds.

7.3.3 Limitations

To create an experiment that would allow us to vary habituation, we opted for a design that was necessarily artificial. Real-world habituation takes place over long periods of time. Security dialogs tend to be viewed in the context of software, one at a time over longer periods of time. Thus, users may behave differently when habituated in a more natural setting. Past usable security work has also found that participants perceive an increased level of safety within a laboratory environment [85], which may make them behave in a less risk-adverse manner. However, our experimental setup allowed us to examine habituation in a manner that corresponded to prior work on attractors [14] as well as prior habituation work [50] in a time- and cost-effective manner.

Since different attractors impose different delays, it was not possible to isolate the habituation

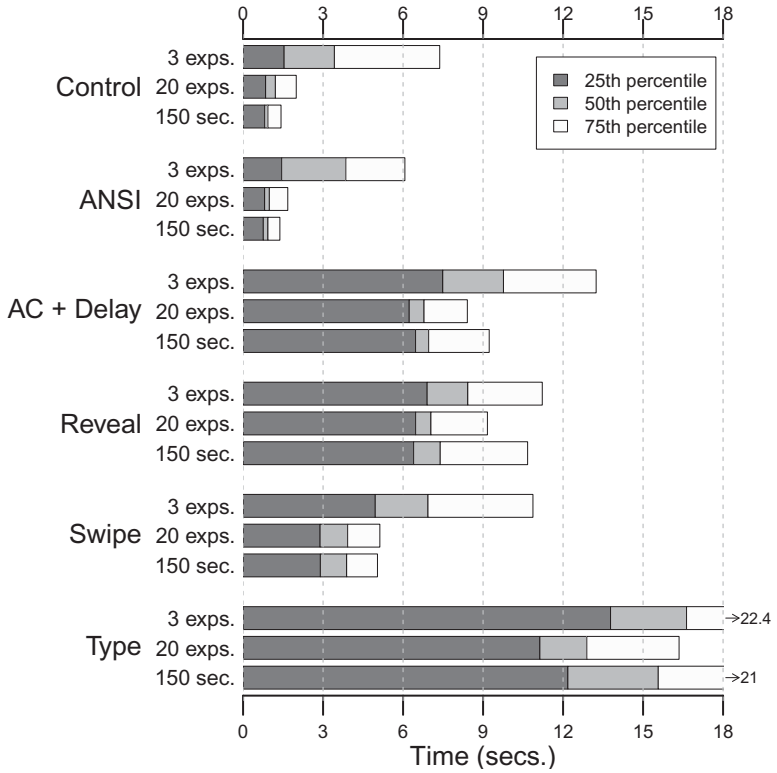


Figure 7.5: We measured the time each participant took to dismiss the last habituation dialog and calculated the median for each condition. We also present the 25th and 75th percentiles. The time spent dismissing a dialog that contained no useful information represents one component of the burden attractors impose on users. (We believe users are less likely to consider time spent reading information salient to a decision to be a burden.)

effects of time and exposure count. Fortunately, this limitation does not appear to impact our conclusions. For the 20-exposure habituation conditions, the *Control* and *ANSI* dialogs required the least amount of time to complete, yielding the shortest habituation time periods, yet they saw the greatest reduction in compliance. In comparison, completing 20 trials took the most time for participants in the *Type* treatment, yet *Type* saw an increased rate of compliance due to habituation.

7.3.4 Conclusions

As expected, habituation reduces attention to the dialog in the experimental task. For the control dialog, habituation reduced the proportion of participants who chose the *No* option by more than a factor of three. We found that using inhibitive attractors significantly reduced the likelihood that participants fail to recognize an instruction contained within a field of a dialog that they had been habituated to ignore.

Furthermore, some attractors are effective in preventing habituation. The *Swipe* condition performed as well in high-habituation conditions as it did in low-habituation conditions, and imposed fairly small usability overhead. During the habituation period 75% of participants learned to dismiss dialogs with the *Swipe* attractor in under five seconds. We expect that much of this overhead

cannot be eliminated, as it includes the time needed to read the information that the attractor is designed to cause them to read. The more arduous *Type* attractor, which imposed a much greater usability burden, actually performed better under high-habituation conditions than low-habituation conditions, perhaps because participants became more motivated to finish the study as the tedious task of retyping text-field contents progressed.

Not all attractors appeared resistant to habituation. The *ANSI* attractor, which used colors to attract attention but did not inhibit participants from choosing the *Yes* option, had no measurable benefit. The *Animated Connector + Delay* and *Reveal* attractors, which delayed the activation of the *Yes* option while showing an animation, performed well under low-habituation conditions but worse in high-habituation conditions, perhaps due to the declining novelty of the animation.

In summary, attractors that require users to take an action that draws their attention, rather than to notice color changes or animation, appear most resistant to habituation in these experimental conditions. Given that users can quickly become habituated to ignore security decisions when previous instances of them have not contained reason for concern, the performance of inhibitive attractors under conditions of artificial habituation is particularly promising.

Chapter 8

Factors that affect user response to security dialogs

In this Chapter I describe two experiments performed to understand two specific aspects of responses to security dialogs. In the first experiment, I explore how increasing the length of text contained in a security decision distracts participants from information presented in the salient field of security dialogs. In the second experiment, I tested two related hypotheses: first, whether reassuring participants about the effectiveness of their antivirus software would increase the proportion of participants who install a suspicious software, and second, whether increasing participants' awareness of the lack of effectiveness of their antivirus would decrease the same proportion. Attractors were not added to the dialogs used in either experiment.

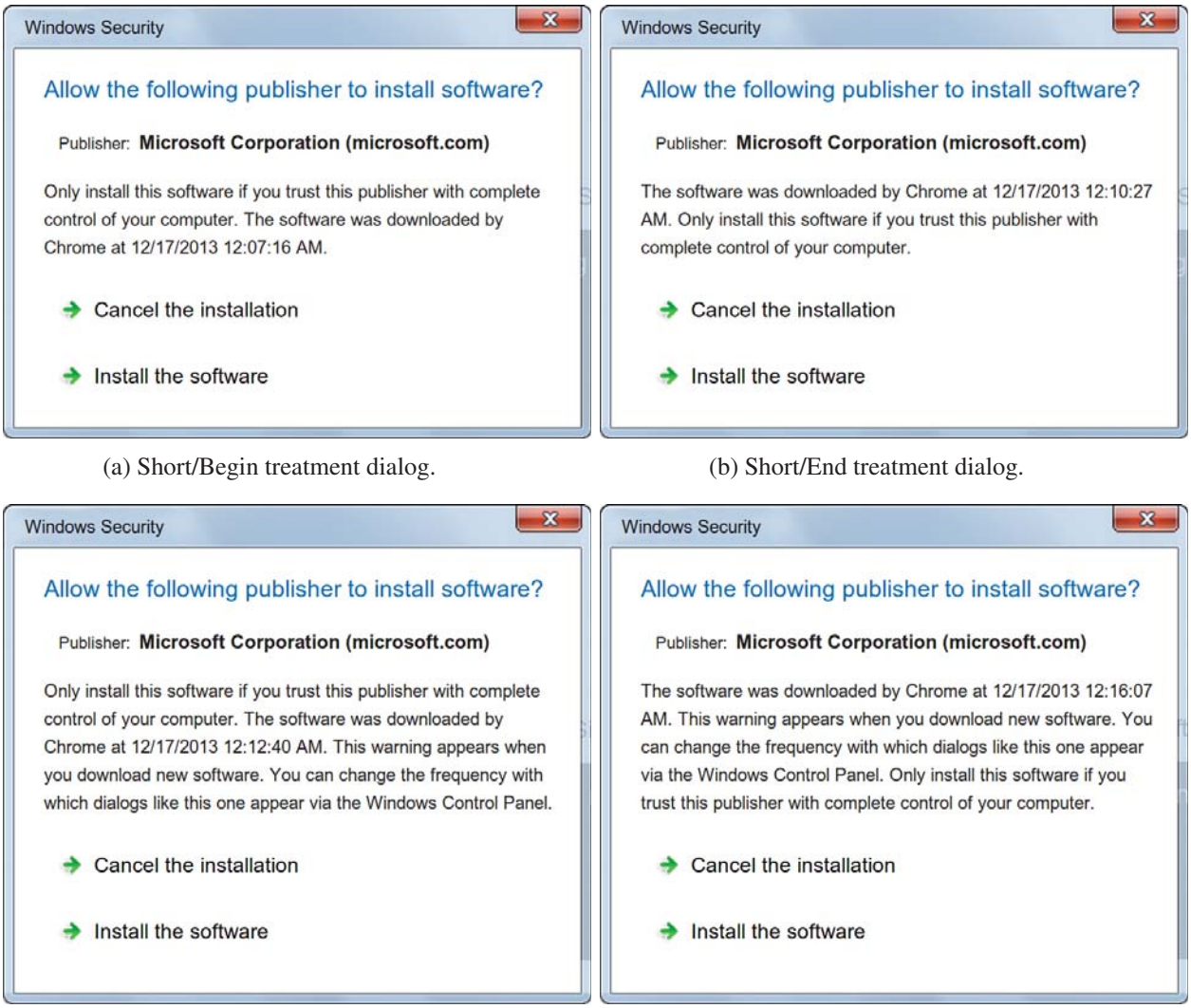
In the first experiment, I found that in a suspicious scenario a) increasing the length of the text in a dialog (up to a point) increases the number of participants who install a likely rogue software, and b) moving security advice from the beginning to the end of the dialog also increases the number of participants who install. I also found that about two thirds of participants were not paying close attention to the dialogs in the suspicious scenario.

In the second experiment I did not find significant differences in installation behavior between dialog conditions. However, participants showed significant differences in their behavior depending on a) the presented scenario (benign or suspicious publisher of a software to be installed), and b) whether they were able to recall correctly the publisher of the software to be installed.

8.1 Experiment 1: Text-length

In this experiment, I explore the effect that security-tangential text added to a dialog may have on attention to a salient field in the dialog. I use both 'Correct Behavior' (that is, installing when the scenario is benign and not installing when the scenario is suspicious) and 'Correct Publisher Recall' (i.e., the proportion of participants who were able to identify correctly the publisher of a software in a dialog) as proxies for attention to the salient field. The assumption is that if a participant does the right thing (that is, Correct Behavior is true), then she must have paid attention to the publisher to make an informed security decision, even if she immediately forgets

This chapter is based on material previously unpublished at the time of writing this chapter.



(a) Short/Begin treatment dialog.

(b) Short/End treatment dialog.

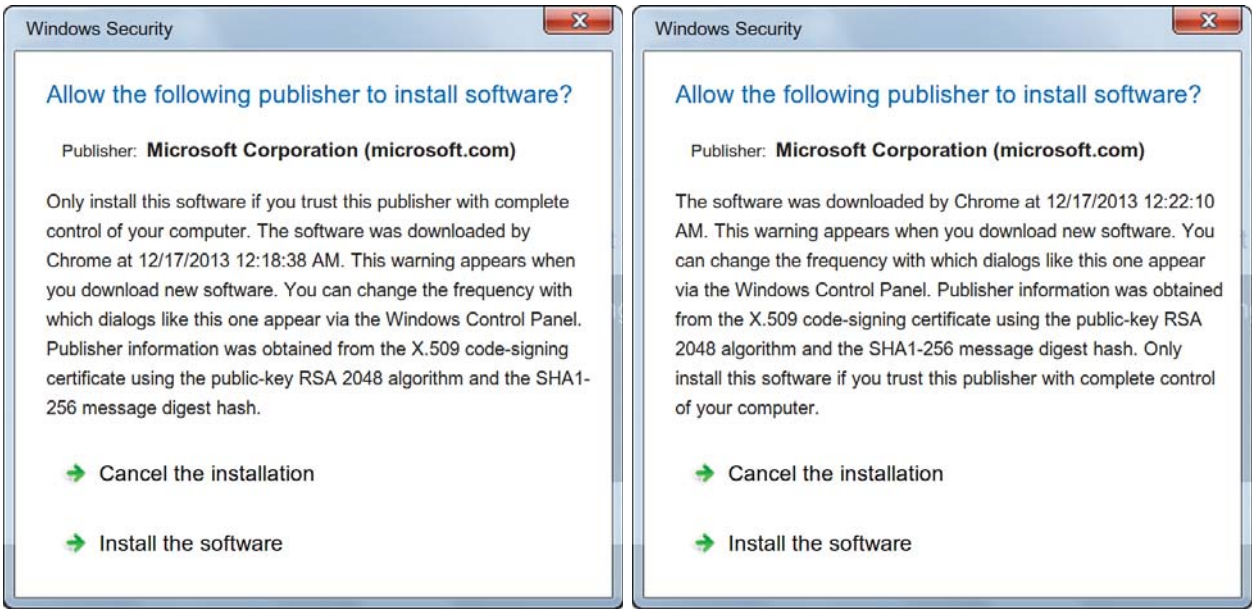
(c) Mid/Begin treatment dialog.

(d) Mid/End treatment dialog.

Figure 8.1: Dialogs for the 'Short' and 'Mid' conditions in the text-length experiment. Only the benign scenario is shown; in the suspicious scenario the salient field was changed to 'Miicr0soft.com'.

this information. Instead, Correct Publisher Recall measures whether a person paid attention long enough to remember it. These two measures may be orthogonal: it is possible to act upon some information and forget it right away; it is also possible to pay attention to a piece of evidence, fail to act upon it due to habituation, and then being able to recall it if asked to do so. Finally, it is also possible to pick the correct option (or the correct behavior) by pure chance.

The dialog used in this experiment was a simplification of the 'Short options' dialog from the Attractors study (Figure 6.9), containing only one sentence delivering security advice: *'Only install this software if you trust this publisher with complete control of your computer.'* (from here on, the 'critical sentence'.) After running a pilot to this experiment, it became evident that using irrelevant or nonsensical text caused participants to question the relevance and trustworthiness of



(a) Long/Begin treatment dialog.

(b) Long/End treatment dialog.

Figure 8.2: Dialogs for the 'Long' conditions in the text-length experiment. Only the benign scenario is shown; in the suspicious scenario the salient field was changed to 'Miicr0soft.com'.

the dialog. Thus, I strove to add text that did not look completely irrelevant, but was tangential to the security advice given in the critical sentence. Similarly, after the pilot I realized that given a long text, it was relevant *where* the critical sentence was within the block of tangential text. Thus, two versions of each dialog were created placing the critical sentence alternatively in the beginning or the end of the text in the dialog.

In this experiment it was necessary to discard the hypothesis that participants would install due to feeling reassured by the text we included; thus, benign and suspicious scenarios were used. In the benign scenario I used 'Microsoft Corporation (microsoft.com)' as the publisher of software that is about to be installed; in the suspicious case, it was 'Miicr0s0ft Corporation (miicr0s0ft.com)'.

8.1.1 Experimental design

A factorial design with 2×2×3 conditions: 2 scenarios (*benign* or *suspicious*), 2 positions of the critical sentence (*beginning* or *end*), and 3 text lengths (*short*, *mid-sized* or *long*), with a total of 12 conditions. The texts used for each condition are presented in Table 8.1, and a summary of the combination of text-lengths and position of the critical sentence are presented in Table 8.2.

As in previous experiments, participants were recruited from Amazon's Mechanical Turk, and the same ruse described in Chapter 5 was used. Each participant was assigned randomly to one of 12 conditions. The experiment had a between-subjects design, where each participant yielded a single data point. After participants clicked on the link presented in the third game page and saw the simulated dialog, they were given an exit survey and were debriefed about the true

	Name	Sentence
CS:	Critical sentence	Only install this software if you trust this publisher with complete control of your computer.
S1:	Tangential sentence 1	The software was downloaded by [browser] at [timestamp].
S2:	Tangential sentence 2	This warning appears when you download new software. You can change the frequency with which dialogs like this one appear via the Windows Control Panel.
S3:	Tangential sentence 3	Publisher information was obtained from the X.509 code-signing certificate using the public-key RSA 2048 algorithm and the SHA1-256 message digest hash.

Table 8.1: Sentences used in the text-length experiment.

Length of text	Position of critical sentence	Sentences used	Figures
Short	Beginning	CS + S1	8.1a
	End	S1 + CS	8.1b
Mid-sized	Beginning	CS + S1 + S2	8.1c
	End	S1 + S2 + CS	8.1d
Long	Beginning	CS + S1 + S2 + S3	8.2a
	End	S1 + S2 + S3 + CS	8.2b

Table 8.2: Conditions used in the text-length experiment, and the corresponding texts used per condition. The plus sign means concatenation of sentences. The acronyms in the column "Sentences used" correspond to those in Table 8.1. The rightmost column shows the Figure where the dialog corresponding to the treatment is shown.

	Short		Mid-sized		Long	
	Begin	End	Begin	End	Begin	End
Benign	109	101	124	98	96	101
Suspicious	113	115	103	102	97	105

Table 8.3: Total number of participants per condition in the text-length experiment.

purpose of the study. Each participant that completed the survey was paid $1.00. Workers that had participated from any prior experiments were excluded from participating in this experiment.

For each participant a binary outcome was observed, depending on whether the participant clicked on 'Install the software' or not. If the simulated gaming page was visited multiple times, the outcome was considered as true if the participant clicked on 'Install' at least once. Then, I defined 'Correct Behavior' as true if a participant saw a benign scenario and installed, or saw a suspicious scenario and did not install.

Similarly to previous experiments, participants were asked what was the publisher in the dialog they just saw through a multiple-choice question. In addition to some distracting choices,

I included the correct options for the benign and suspicious scenario ("microsoft.com" and "micr0s0ft.com", respectively), plus an 'Other' option to allow participants to provide any additional information they wanted to. Free responses to the 'Other' choice were coded.

I measured the proportion of participants who were able to identify the publisher correctly ('Correct Publisher Recall' from here on), based on participants' answers to the question "What was the name of the publisher of the software to be installed? (if you are not sure, please provide your best guess)" (see the full question in Appendix F.2).

Finally, I measured the response time for each participant: that is, the time span between when the dialog appeared for the first time on a participant's screen and when the participant responded to the dialog in any way, e.g., by clicking on one of the options or on the close button in the top right corner of the dialog.

Text-length was coded as an ordinal variable with three levels: Short, Mid, and Long. Position of the critical sentence (or simply 'position') was coded as a categorical variable with two levels: Begin and End. Finally, scenario was also coded as a categorical variable with two levels: Benign or Suspicious. I conducted a logistic regression analysis in order to test the following null hypotheses:

H1. Changes in scenario do not have effects over Correct Behavior.

H2. Changes in text-length do not have effects over Correct Behavior.

H3. Changes in position do not have effects over Correct Behavior.

H4. Changes in scenario do not have effects over Correct Publisher Recall.

H5. Changes in text-length do not have effects over Correct Publisher Recall.

H6. Changes in position do not have effects over Correct Publisher Recall.

8.1.2 Results

Participants and install rates

N=1,264 participants were recruited between October 8 and November 14, 2013. Participants were 29.9 years old in average (σ =9.3 years old), 52% of them were male and 46% were female, 76% of them were caucasian, and the top two reported occupations were 'student' (20%) and 'unemployed' (17%). 27% of participants reported to have knowledge of a programming language. The number of participants per condition are presented in Table 8.3.

I present the install rate per condition in Figure 8.3. In general, a higher proportion of participants installed in the benign scenario than in the suspicious scenario, as expected.

Effects of studied factors on Correct Behavior

In Table 8.4a I present the results of a logistic regression with Correct Behavior as binary outcome. Of the studied variables, only the scenario had a significant main effect on correct behavior; that is, participants who saw the suspicious scenario were significantly more likely to take the correct action. This provides evidence to reject the null hypothesis in H1: a change from benign (baseline) to suspicious scenario has a significant, positive effect on Correct Behavior ($e = 0.4133$, $p < 0.0004$). In other words, participants who saw the suspicious scenario were significantly more

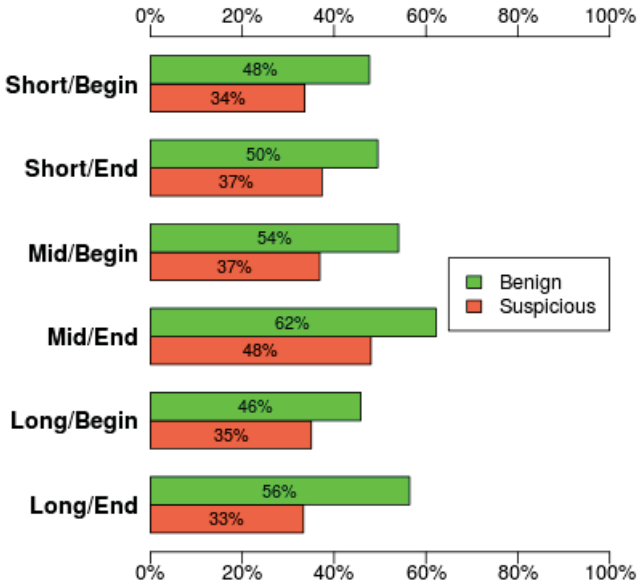
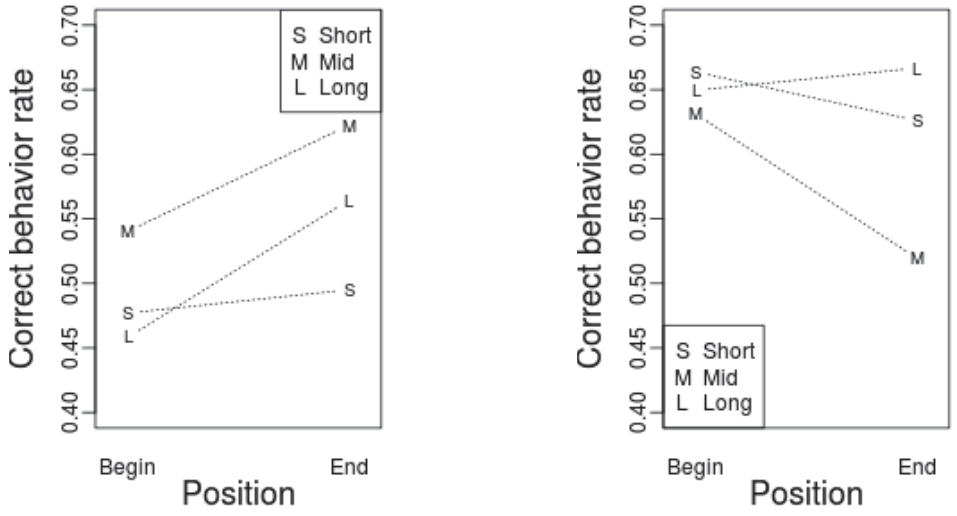


Figure 8.3: Install rate per condition in the text-length experiment.



(a) Benign scenario.

(b) Suspicious scenario.

Figure 8.4: Interaction plots showing Correct Behavior for the benign (left) and suspicious (right) scenarios.

likely to 'act correctly' (i.e., not installing) than participants who saw the benign scenario (i.e., to install).

Neither text length nor position had significant main effects over correct behavior. However, when second-order interactions are included in the regression (see Table 8.4b), two effects can be observed:

1. There is a significant positive interaction between the suspicious scenario and a quadratic term of the length ($e = 0.5336$, $p = 0.0071$). That is, increasing the length of the text in

Variable	Estimate	z value	p-value
Scenario *(suspicious)*	0.4133	3.61	<**0.0004**
Position of critical sentence *(end)*	0.0425	0.37	0.7106
Length *(linear)*	0.0562	0.56	0.5731
Length *(quadratic)*	-0.0079	-0.08	0.9364

(a) Logistic regression, main effects over Correct Behavior.

Variable	Estimate	z value	p-value
Scenario *(suspicious)*	0.6481	3.99	<**0.0001**
Position of critical sentence *(end)*	0.2807	1.74	0.0817
Length *(linear)*	-0.0345	-0.20	0.8417
Length *(quadratic)*	-0.3281	-1.99	**0.0467**
Position *(end)* * Scenario *(suspicious)*	-0.4627	-2.01	**0.0448**
Position *(end)* * Length *(linear)*	0.2092	1.04	0.2987
Position *(end)* * Length *(quadratic)*	0.1273	0.64	0.5213
Scenario *(suspicious)* * Length *(linear)*	-0.0306	-0.15	0.8794
Scenario *(suspicious)* * Length *(quadratic)*	0.5276	0.1985	**0.0079**

(b) Logistic regression, second-order effects over Correct Behavior.

Table 8.4: Results of logistic regressions of independent variables over Correct Behavior in the text-length experiment. The top table includes main effects only, while the bottom table includes second-order effects.

the suspicious scenario has a positive quadratic effect on correct behavior. In other words, while the length of the text in tested dialogs has no main effect over correct behavior, the combination of both longer texts and a suspicious scenario have a reinforcing effect on each other that results in more participants 'acting correctly' (i.e., not installing) than it would have resulted from the addition of the effects of both factors. The effect is quadratic, so when going from Short to Mid correct behavior decreases, but when going from Mid to Long correct behavior increases. One possible explanation for this is as follows: a *slight increase* in text length (below some unknown threshold) indeed decreases correct behavior; however, when the length of the text goes beyond that threshold, participants become more suspicious and the text length ceases to be a *distracting* factor.

2. There is a significant negative interaction between the suspicious scenario and the position of the critical sentence ($e = -0.4653$, $p = 0.0436$). In other words, while the position of the critical sentence has no main effect on correct behavior, moving the critical sentence to the end of tested dialogs in a suspicious scenario makes less likely that participants 'act correctly' (i.e., that participants do not install).

There are no significant third-order interactions in the studied models.

The previous second-order interactions have in common a suspicious scenario, and involve both position and text length, the latter variable not under a linear but a quadratic form. This provides evidence to reject the null hypotheses in both H2 and H3.

In Figure 8.4, I plotted the same information as in Figure 8.3. If we examine the lines for the Short and Mid treatments first, we will notice that their slopes and spatial relationship are similar

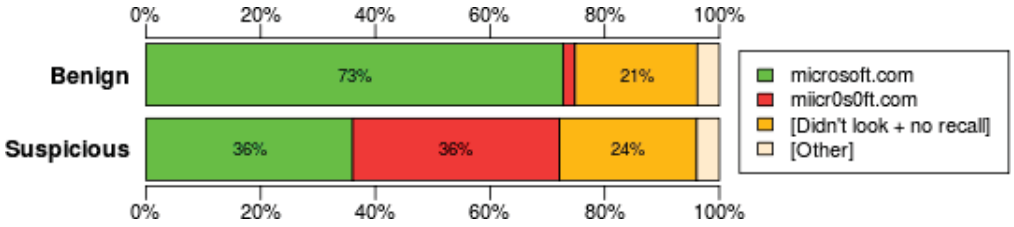


Figure 8.5: Proportion of participants who answered correctly the question "What was the name of the publisher of the software to be installed? (if you are not sure, please provide your best guess)" in the text-length experiment.

across scenarios, unlike the lines for the Long treatments. The line for the Long treatment in the benign scenario has a similar slope to the Mid line, although with lower levels of installing rate; whereas in the suspicious scenario, the line is almost horizontal, and its level is similar to that of the corresponding Short line.

Variable	Estimate	z value	p-value
Scenario *(suspicious)*	-1.5591	-12.75	<**0.0001**
Position of critical sentence *(end)*	0.0140	0.12	0.91
Length *(linear)*	-0.0694	-0.66	0.51
Length *(quadratic)*	0.0509	0.48	0.63

(a) Logistic regression, main effects over Correct Publisher Recall.

Variable	Estimate	z value	p-value
Scenario *(suspicious)*	-1.5068	-8.81	<**0.0001**
Position of critical sentence *(end)*	0.0807	0.45	0.66
Length *(linear)*	-0.1771	-0.94	0.34
Length *(quadratic)*	-0.0948	-0.52	0.60
Position *(end)* * Scenario *(suspicious)*	-0.1081	-0.44	0.66
Position *(end)* * Length *(linear)*	0.1248	0.60	0.55
Position *(end)* * Length *(quadratic)*	-0.0819	-0.38	0.70
Scenario *(suspicious)* * Length *(linear)*	0.0869	0.41	0.68
Scenario *(suspicious)* * Length *(quadratic)*	0.3457	1.61	0.11

(b) Logistic regression, second-order effects over Correct Publisher Recall.

Table 8.5: Results of logistic regressions of independent variables over Correct Publisher Recall in the text-length experiment. The top table includes main effects only, while the bottom table includes second-order effects.

Effects of factors on Correct Publisher Recall

In Figure 8.5, I present the answers to the question "What was the publisher?" The correct answer depends on the scenario: 'microsoft.com' for benign, and 'miicr0s0ft.com' for suspicious. The proportion of Correct Publisher Recall in Figure 8.5 drops to a half in the suspicious case, and

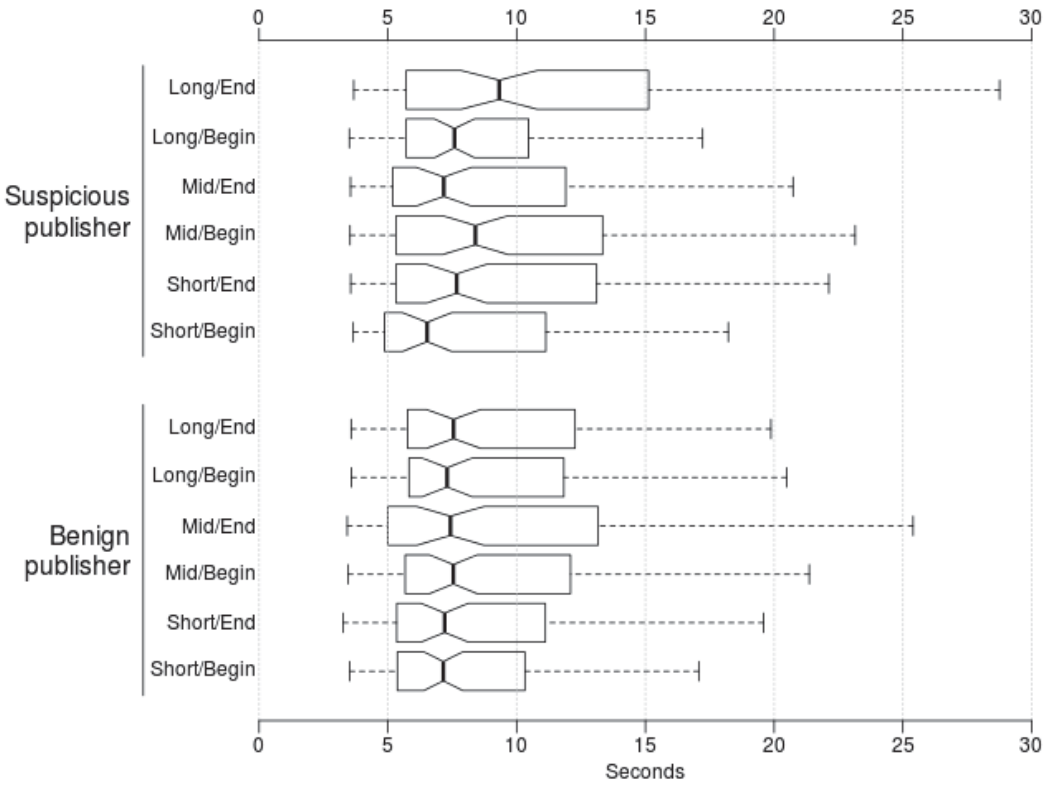


Figure 8.6: Response time per condition in the text-length experiment.

does not change much across conditions. One possible reason for this behavior is that a large part of participants were not actually paying attention, and responded to the dialogs in an automatic fashion. When participants reached the simulated dialog, most of them perceived something that looked liked a Microsoft dialog, and answered the question above accordingly (i.e., "It's a Microsoft dialog.") However, this answer turns out to be correct in the benign scenario and incorrect in the suspicious scenario. An inspection of Figure 8.5, along with an analysis of the response time in Figure 8.6 suggests that this may have happened to approximately a third of our participants. There is about a fourth that did not have any recall of the publisher and admitted so, and about a 5% of participants who guessed wrong (i.e., they answered 'Other', and their answer could not be classified as either 'microsoft.com' or 'miicr0s0ft.com'.) That leaves us with approximately a third of participants who might have been paying closer attention.

I conducted a logistic regression to determine whether any factors influenced the Correct Publisher Recall. Results are presented in Table 8.5. Only the scenario has a significant effect on Correct Publisher Recall ($p < 0.0001$), providing evidence to reject the null hypothesis in H4. There are neither second- nor third-order effects on the outcome. Thus, there is no evidence to reject the null hypotheses in H5 and H6.

Finally, there is a strong correlation between Correct Behavior and Correct Publisher Recall ($\chi^2(1) = 27.98, p < 0.00001$).

Response time

In Figure 8.6 we present the response time per condition; that is, the time span between when a dialog pops up on the screen, and when the user clicks on any button within the dialog. For users who loaded the page two or more times (and watched two or more dialogs) only the response time of the first dialog was considered. Median response time for the six conditions in the benign scenario is 7.3 seconds (25th and 75th percentiles are 5.5 sec and 11.8 sec, respectively), and an omnibus Kruskal-Wallis test is not significant ($\chi^2(5) = 2.74$, $p = 0.739$).

In the suspicious scenario, participants seem to take longer to respond. In the Long/End condition participants took the longest (25th, 50th, and 75th percentiles are 5.7 sec, 9.3 sec, and 15.1 sec, respectively). An omnibus Kruskal-Wallis test for the suspicious scenario is significant ($\chi^2(5) = 12.51$, $p = 0.028$).

This provides some evidence to support the notion that approximately two thirds of our participants were not paying close attention to the dialogs, and in most cases their response was probably the consequence of some level of already-existing *habituation*. Nevertheless, this is moderated by the fact that install rates were consistently 11 to 13% higher in the benign cases than in the corresponding suspicious cases, and that the overall median response time was about 7.5 seconds – a relatively long time for a response to a dialog.

8.1.3 Conclusion

I conducted an experiment to determine whether increasing amounts of text within a security dialog would distract users exposed to the dialog from paying attention to salient fields in the dialog. I used two variables as proxies for attention: Correct Behavior and Correct Publisher Recall. I found that a) increasing the length of text in a security dialog (to a point) decreases Correct Behavior in the suspicious scenario, and b) placing security advice in the end of a dialog also decreases Correct Behavior in the suspicious scenario. However, except for the scenario, none of the studied variables had significant effects on Correct Publisher Recall. This provides conflicting evidence about the role of the studied factors over attention: while they influenced participants' behavior, data suggests that these decisions were made automatically, without a minimal awareness of the decision just made.

8.2 Experiment 2: Antivirus software

In previous experiments, participants were asked why they chose to install software that appeared to be malicious. A fraction of them consistently reported that they had antivirus software installed, and that they trusted their software to protect them from malicious software. For example, the following answers came from participants in the text-length experiment:

> **Q**: In previous questions, we asked you "What did you do when the window appeared on your screen?", and your answer was "I clicked on 'Install the software'." Later, we asked you if you recalled the publisher of the software, and your answer was ___. Could you please explain why you decided to install the software even after recognizing the odd-looking publisher name?

(a) Control dialog, benign scenario.

(b) Control dialog, suspicious scenario.

(c) Risk-aversion dialog, benign scenario.

(d) Risk-homeostasis dialog, benign scenario.

Figure 8.7: Dialogs used in the anti-virus experiment. On the top row, the control dialogs are shown; while on the bottom row the 'Risk-aversion' and 'Risk-homeostasis' conditions are shown. Only the benign scenario is shown for the bottom row.

A1: "I assumed since it was being linked through this survey it would be safe, and I considered my anti viruses and anti malware software would block anything harmful."

A2: "I have a great antivirus if anything was wrong with it my antivirus would delete it."

A3: "I have a pretty good anti-virus software which prompts me when malicious software is being downloaded. I figured if it was not safe I would get a prompt."

A4: "I just didn't think about it. I have virus software and assumed it would let me know if it was something bad."

Variable	All ppts.	Ppts. w/ AV	Ppts. w/o AV	Sig. Diff.?
Number of participants (N)	1,570	1,218	352	
Gender (male)	51%	53%	47%	FET:
(female)	47%	46%	52%	$p = 0.17$
Age (mean)	30.1 y.o.	30.6 y.o.	28.3 y.o.	KW: $\chi^2(1) = 14.9$,
(std. dev.)	9.5 y.o.	9.8 y.o.	8.2 y.o.	$p = 0.00011$
Ethnicity (top reported)	Caucasian (76%)	Caucasian (77%)	Caucasian (71%)	$\chi^2(7) = 8.79$, $p = 0.27$
Occupation (top reported)	Student (20%)	Student (20%)	Student (19%)	$\chi^2(15) = 10.74$, $p = 0.77$
Has knowledge of prog. lang.? ('Yes')	25%	27%	20%	FET: $p = 0.02$

Table 8.6: Demographic data of participants in the anti-virus experiment. The left column ('All ppts.') presents data for the whole dataset; the middle column presents data for participants who reported to have antivirus software installed in the computer they used to answer our survey; the rightmost column shows data for participants who reported to not have antivirus software installed.

Variable	Estimate	z value	p-value
Risk-aversion treatment	0.1246	0.839	0.402
Risk-homeostasis treatment	0.0173	0.116	0.907
Scenario (*suspicious*)	1.2230	8.837	<**0.0001**
Care for computer (*true*)	0.1901	1.338	0.181
Recall of publisher (*correct*)	1.1807	8.537	<**0.0001**

Table 8.7: Logistic regression over 'correct behavior' in the anti-virus experiment.

	Control	Risk-aversion	Risk-homeostasis
Benign	204	215	202
Suspicious	203	195	199

Table 8.8: Total number of participants per condition in the anti-virus experiment. Only those participants who reported having antivirus software installed are included.

A5: "My antivirus would block it if it was damaging"

A6: "My virus protection did not go off"

I decided to test the idea that computer users may take additional risks when they trust in their antivirus software to stop any harm. Using the same method described elsewhere, I designed an experiment in which participants would be suggested to download and install an application through a security dialog. In this dialog, participants would be given two conflicting signals: first, that their antivirus software checked the application and found no malware on it; and second, that the publisher of the software was the odd-looking 'miicr0s0ft.com'. I expected that participants would give priority to the first signal over the second, and install more often than in a control condition where such signals were not presented. I also tested the opposite idea: if participants were

told that their antivirus software could not run, would that make them behave more cautiously?

I modified the 'Short options' dialog used in the Attractors study (Figure 6.9) by adding a second salient field to accommodate information about simulated antivirus software (see Figures 8.7a and 8.7b). Starting from this dialog, I designed two conditions:

1. A *Risk-aversion* condition, wherein the dialog contained the following advice in the second salient field: "antivirus check not complete: database currently unavailable" (see Figure 8.7c). I expected that participants in this treatment would install less frequently than participants seeing the control dialog.

2. A *Risk-homeostasis* condition, wherein the dialog contained the following advice: "antivirus check complete: no known viruses detected" (see Figure 8.7d). I expected that participants who received this dialog would feel confident and reassured, and thus they would install more frequently than participants who saw the control dialog.

As before, both benign and suspicious scenarios were used. I expected that neither risk-aversion nor risk-homeostasis conditions would affect participants' responses in the benign scenario, while I expected that participants would install more frequently in the risk-homeostasis treatment and less frequently in the risk-aversion treatment, both compared to the control/suspicious treatment.

8.2.1 Experimental design

I designed a between-subjects experiment with a 2×3 factorial design: 2 scenarios (*benign* or *suspicious*), and 3 treatments (*control*, *risk-aversion*, and *risk-homeostasis*), for a total of 6 conditions. Each participant was assigned one condition randomly. As before, participants were recruited from Amazon's Mechanical Turk. I used the same methodology described in Chapter 5.

Participants were asked to evaluate three games, the third one of which displayed a simulated installation dialog. After a participant responded to this dialog, she was directed to questions about her response. Finally, participants answered demographic questions and were debriefed about the true purpose of the study. Each participant who completed the survey received $1.00. As before, workers that participated in any of our previous experiments were banned from participating in this one.

As before, I constructed a binary outcome (Correct Behavior) based on whether each participant picked 'the right option' in each scenario (i.e., installing in the benign scenario and not installing in the suspicious scenario). If a participant saw more than one dialog, the outcome was positive if the participant installed at least once in the benign scenario, and never installed in the suspicious scenario. As before, participants were asked whether they could recall the publisher of the software being identified in the security dialog. I used their answers to construct another binary outcome: correct recall of the publisher.

In this experiment it was important to understand whether participants had an antivirus software installed in the computer they were using to participate in the study, and whether they cared for such computer. If a participant had no antivirus software, messages displayed by the security dialogs would be absurd. Since it is technically hard to detect automatically whether such software was installed in participants' computers, I included the following question in the exit survey: "Do you have antivirus software installed on the computer you used to perform this experiment?"

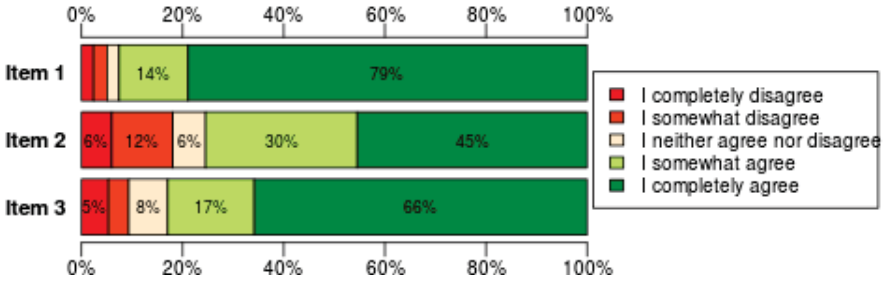


Figure 8.8: Answers to the items measuring 'care for the computer' that participants are using to answer the survey. Used items are 1: "It would be a major hassle for me if the computer I am using for this study got infected with a virus.", 2: "If the computer I am using for this study got infected with a virus, that would take a lot of effort to fix.", and 3: "It would be a waste of time if the computer I am using for this study got infected with a virus.".

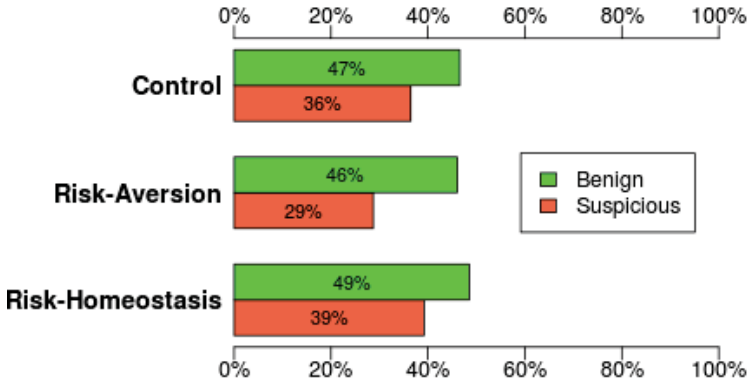


Figure 8.9: Install rate per condition in the anti-virus experiment. Only those participants who reported having antivirus software installed are included.

If participants answered 'Yes', we asked them to indicate in a free response what antivirus they had installed. Although initially I planned to include only the data of participants who answered affirmatively this question, this would introduce a systematic bias: participants who had antivirus installed in their computer are probably different than those who do not. I repeated all analyses both for the whole group, and for those participants who reported to have antivirus software installed.

Finally, I designed a set of Likert-type questions to determine whether participants cared for the computer they were using to take the exit survey. I did not want to overwhelm participants with many questions in an already-long study; so I decided to keep only three questions out of a larger set. The question was: "Please indicate how much do you agree with the following sentences." The sentences to be evaluated by participants were:

1. "It would be a major hassle for me if the computer I am using for this study got infected with a virus."

2. "If the computer I am using for this study got infected with a virus, that would take a lot of effort to fix."

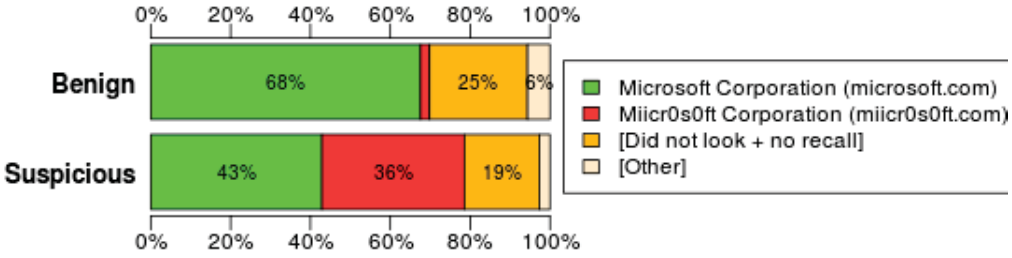


Figure 8.10: Proportion of participants who answered correctly the question "What was the name of the publisher of the software to be installed? (if you are not sure, please provide your best guess)" in the anti-virus experiment.

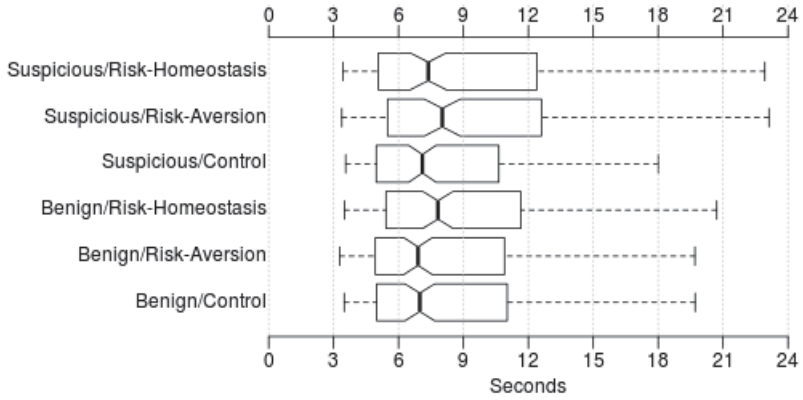


Figure 8.11: Response time per condition in the anti-virus experiment. Only participants who claimed to have antivirus software installed are included.

3. "It would be a waste of time if the computer I am using for this study got infected with a virus."

Each of the items above was presented as 5-points Likert-type scale, from 'I completely disagree' to 'I completely agree'. The outcome was defined as positive if all three questions were answered as 'I agree' or 'I completely agree.'

I measured the internal consistency of the set of questions in a pilot previous to the experiment and obtained a Cronbach's alpha metric close to 70%. Although I strove to obtain a high level of reliability as measured by the Cronbach's alpha metric, the actual result was lower than what we obtained in the pilot, and lower also than the usual standard that is required for psychological research and medical studies [89]. Two reasons may be the low number of items (three), and that I may be measuring more than one concept with these three questions (i.e., a problem of lack of unidimensionality). It is likely for example that the questions actually measured whether participants anticipated that they would be inconvenienced if their computer got infected. Figure 8.8 shows the aggregated answers for the three items in this construct. As seen in the Figure, item 2 deviates from items 1 and 3. Although it is possible to omit item 2, I decided to keep the whole set because the concept I am trying to measure was very specific to this experiment, and the level of needed internal consistency was not actually very high. This metric should be reviewed in future

studies: questions should be added appropriately, and the unidimentionality of the scale should be ensured.

8.2.2 Results

N=1,570 participants were recruited between October 8 and December 6, 2013[1]. 1,218 of these participants reported in the exit survey to have antivirus software installed. Table 8.6 contains demographic data for the whole dataset and the subgroup mentioned above. Although there were significant differences in age and knowledge of a programming language (the group that does not have antivirus software installed is slightly younger and less technically-savvy), data for both groups is similar and the differences have little practical importance. From here on, unless indicated otherwise I present data only for the group who reported having antivirus software installed.

Figure 8.9 shows the install rate per condition. Install rates show little variation in the benign scenario (46 to 49%), as expected. In the suspicious scenario, the install rate shows a greater variation: 36% in the control condition, 29% in the risk-aversion condition, and 39% in the risk-homeostasis condition. However, none of these differences is significant. Table 8.7 contains a logistic regression performed over Correct Behavior. Neither the Risk-aversion nor the Risk-homeostasis conditions are significantly different from the control condition. The variable 'Care for computer' has no significant effect on correct behavior either. However, both scenario and correct recall of the publisher show significant positive effects over the outcome. That is, participants who saw a suspicious publisher were significantly more likely to 'behave correctly' ($p < 0.0001$) compared to participants who saw the benign publisher, and participants who later recalled the publisher correctly were also significantly more likely to behave correctly ($p < 0.0001$).

As Table 8.8 shows, the number of participants per condition is relatively high in this experiment. An estimation of the effect size of install rates between Control/Suspicious and Risk-aversion/Suspicious yields a Cohen's h value equal to 0.17, which can be considered small [2]. As a consequence, even if there is an effect produced by the additional information provided in the risk-aversion and risk-homeostasis conditions, the effect is too small to be of any practical importance. In other words, giving participants relevant contextual information about the results of an action performed by their antivirus software has no practical effects on participants' correct behavior.

Figure 8.10 contains participants' answers to our request to recall the publisher. Similarly to what happened in the text-length experiment, about 60% of participants might have been not paying close attention to our dialogs in the suspicious scenario (43% that recalled incorrectly the publisher, plus about 20% that admitted not having any recall). Participants also had a similar distribution of time to respond to all benign conditions (see Figure 8.11). The Suspicious/Control condition was not different from the Benign/Control in terms of response time. Across the suspicious scenario, participants took slightly longer to respond to risk-aversion and risk-homeostasis than to the control dialog, although only the difference between control and risk-aversion was slightly significant (Kruskal-Wallis Rank sum test, $\chi^2(1) = 5.108$, $p = 0.024$, not corrected for multiple testing).

[1] Data collection was simultaneous to the text-length experiment

[2] Cohen suggested that 0.2 should be considered a small effect, 0.5 a medium effect and 0.8 a large effect [73, p. 373]. The power of the current experimental setup ($1 - \beta$) was estimated in 38% for an $\alpha = 0.05$; in order to reach 80% of statistical power, approximately 380 more participants in each condition would be needed; that is, approximately 2,280 additional participants in total.

8.2.3 Conclusion

Overall, participants behaved 'reasonably' from a security perspective. Those who were able later to recall correctly the publisher were significantly more likely to pick the right option depending on the scenario (i.e., installing in the benign scenario, and not installing in the suspicious scenario). Although about two thirds of them were not paying close attention to the presented dialogs in the suspicious scenario, based on their recall of the publisher and the time they took to respond to our dialogs, those who saw the suspicious scenario were significantly more likely to do the right thing (i.e., not installing) than those who saw the benign scenario (in which case the right thing to do was to install).

Interestingly, offering information about the results of an action performed by the antivirus software had in practice no effect on participants' behavior. This disproves the utility of a suggestion that my co-authors and I made during the mental model study (see Section 3.3). This might be due to the 'publisher signal' being strong enough to override the usefulness that the additional antivirus information might offer.

8.3 Overall conclusion

I performed an experiment to learn whether two specific factors – length of text within a security dialog, and the position of a fixed security advice within a dialog – affected computer users' response to a request to install a possibly rogue software. I found that in a suspicious scenario, both the length of the text in a dialog (up to a point) and the position of the security advice in the dialog have a significant effect on participants' correct behavior (that is, not installing). I also found that these factors may be affecting participants' behavior (that is, what they did when presented with a dialog), but not necessarily their attention to the presented information. This suggests that participants' decisions were made 'automatically', without much contextual awareness.

I performed a second experiment to learn whether contextually relevant antivirus information offered through a dialog changed participants' behavior: installing or not a possibly rogue software. I found that such information had no practical effect on participants' installing behavior, possibly because the signal delivered through the publisher was strong enough to override the presented antivirus information.

Chapter 9

A brief analysis of software vendors' and users' decisions

Software vendors create and distribute software, which is a remarkably complex product. Its usage has created new risks (e.g., malware infection, phishing) or transformed old risks (e.g., cyber-stalking) in a way that is often difficult to anticipate. In order to prevent users from exposing themselves to those risks that can be anticipated, vendors implement security dialogs in their products. Programmers and software designers face two problems: when to display a dialog, and what information to include in a dialog once it is displayed.

In this chapter I describe three incentives that push vendors to include more dialogs than necessary. Then, I describe how to estimate the number of socially optimal dialogs, using some of the findings presented throughout this thesis as inputs. Finally, I make some recommendations to incentivize companies to decrease the number of dialogs they present to users in their products.

9.1 Vendor incentives to show security dialogs

There are at least three incentives that vendors face to implement security dialogs in their products: uncertainty in the anticipation of risks, a tendency to avoid liability, and user preferences. As a result, vendors tend to implement more dialogs than necessary, from the point of view of the consumer.

9.1.1 Risk anticipation uncertainty

In the physical world, a warning is usually devised by a product's manufacturer or by an agency enforcing a health or safety regulation (see Section 2.1.2). Most risks are known, or assumed to be known, by the manufacturer [58]; companies collect usage data from misuse cases or accidents with their products to better understand those risks that cannot be anticipated. New warnings are then created, or existing ones are modified, to accommodate new information [59].

Similarly, vendors try to anticipate the risks that arise from software usage, and implement security dialogs for those risk scenarios to which they know users will be exposed. Unlike physical products, anticipating risks in software usage scenarios can be very difficult due to both the lack of materiality of software and the networked nature of most software in use today (e.g., browsers and

email clients). Given this complexity and the uncertainty of the risks involved, it is reasonable to assume that vendors prefer to implement and show a security dialog rather than trying to anticipate all possible risk scenarios.

9.1.2 Liability avoidance

A second reason vendors may choose to implement security dialogs is to avoid the threat of litigation.

Whenever a product causes harm to its consumers, the manufacturer may be held liable for the harm. Liability is part of United States legal framework, and is rationalized on the basis of economic efficiency: "the law tries to incent manufacturers to take into account the potential harms that their products may cause to the public in the most efficient manner possible" [8]. Products liability in the United States developed as a result of changes in privity law, when implicit contract obligations between craftsmen producers and consumers were no longer applicable to the mass production that emerged as a consequence of industrial growth [8, p. 207].

Manufacturers can be held liable for harm based on the theory that, as experts about their products, they are in a better position than consumers to determine if there are any risks associated with the use or consumption of the product [58], so they have the ethical duty to warn about the risk. Doing so is "consonant with the ethical principles of autonomy and beneficence" [12, p.11]. Warnings have been critical in liability suits in the United States, where a defendant's culpability is frequently determined based on his or her lack of attempt to warn about a product's hazards [35].

When a potentially harmful but useful product (e.g., a knife, or a hammer[1]) is produced, and the manufacturer has to assume part of the costs of the harm when there is harm, the cost of producing the good increases and the supply of the good decreases. A manufacturer that does not want to decrease its revenues should either improve its product to make it less dangerous, or train its consumers to use the product in a less harmful way. For example, a knife manufacturer may change the knife design to make it less likely to cut oneself by adding a safeguard, or it may include a pamphlet with safety instructions along with each knife. Many products are inherently dangerous but tremendously useful to society, and should not be banned from markets despite the fact that accidents are likely to occur from use of the product.

Concerning physical warnings, there are two conflicting incentives for product manufacturers. On the one hand, companies want to transfer liability to the consumer by including as many warnings as possible for every conceivable risk, even when in some cases (e.g., strict liability) a warning will not transfer liability completely to the consumer. On the other hand, there is some evidence that warnings that accurately reflect risks with physical products may decrease profit [12, 23] [2]. Given that including warnings for some risks is mandatory, companies have modified warnings in the past to avoid decreased profitability. Authors have documented industry attempts to soften mandated warnings in the tobacco, automobile and pharmaceutical industries. They reported two strategies used by companies: weakening a warning, and producing media

[1] I do not include in this categorization products like tobacco and alcohol, whose consumption very often policy makers seek to decrease to zero [12].

[2] Not all warnings that accurately represent a risk may actually decrease sales. Consider, for example, two brands of rat poison that, except for a health warning, are in all other aspects the same. A potential consumer may prefer the brand with the strongest warning in the belief that that product will be more effective for the intended use [23].

to misrepresent the risk to make it appear as less risky than it actually is [12, 23]. The second strategy has been used when an accurate warning may decrease profit, for example, with some types of airbags that may kill a car's occupants in low-speed collisions, and with tobacco in the first half of the 20th century [12].

Such conflicting incentives do not exist in software markets. In general, software companies avoid liability in part by putting themselves under a different legal regime: instead of selling products, they typically sell licenses to use their products (so-called End User License Agreements, or EULAs), which include a long list of caveats and exceptions [25, 90]. Given that a) there are no regulations mandating the inclusion of computer security dialogs in software, and b) vendors have incentives to shift liability to the consumer (as evidenced by their use of EULAs), vendors are free to warn the user as often as they want, in the way they prefer. This results in an excessive use of security dialogs, probably beyond what is socially optimal.

9.1.3 User preferences

A third reason software vendors choose to implement security dialogs is related to user preferences and user perception of security and usability of software.

As in any other industry, software vendors' main interest is profit. Although not all software is actually sold (e.g., most web browsers today are offered for free), software companies must have some kind of business model. For example, Google offers most of their software services for free, using these services as an integrated platform to show advertisements [3]. In general, profit depends on demand, and demand depends, among other factors, on:

The existence of substitute products. In some specific cases (e.g., web browsers), the cost of switching to a substitute product is negligible; as a result, many company decisions are based on user preferences and perceptions, to the detriment of user security.

Consumer perception of product quality and safety. Since a completely unusable or notoriously unsafe software simply would not be bought or installed, vendors must have an interest in making their products usable and secure. The question most vendors must answer is not whether to design usable software (in particular, how to design usable security dialogs), but how best to do so.

Two examples may help to illustrate the points above:

1. When users visit websites with incorrectly configured SSL certificates, the browser has three options: to show the website, block the website, or show a warning and let the user decide what to do. In general, browser vendors prefer to show a certificate warning rather than just blocking websites with incorrectly configured SSL certificates because if they block sites with misconfigured certificates, users may think the browser is not working or defective and switch to another browser. On the other hand, if they show websites with incorrect SSL certificates, the browser may be perceived as insecure and a number of users may also switch to another browser. By showing a warning, they transfer the problem to the user.

[3] 91% of Google's 2013 total advertising revenues came from digital advertisements[39].

2. In some cases, browsers may gather reasonably sound evidence that a website is a phishing scam. Although the safest action would be to simply block the website (i.e., not let the user open it), browser vendors prefer to show a warning instead because accidentally blocking a non-phishing website may originate potentially expensive lawsuits. Again, showing a warning transfers the problem to the user.

9.2 User costs due to habituation

In the previous section I identified three existing incentives for software vendors to show a large number of security dialogs. Why is a large number of dialogs a problem?

Each dialog containing a warning that does not bring negative consequences for the user is perceived as a false alarm (see Section 2.1.5), decreasing the level of attention to all similar dialogs. Many security dialogs are not very different from each other; and in lab studies, users that have been exposed to made-up security dialogs have stated that they had seen those dialogs before [87]. I described in Chapter 7 how a low number of exposures to similar dialogs is sufficient to decrease rapidly the time that computer users spend responding to each subsequent dialog. Thus, a large number of dialogs leads to habituation to similar dialogs (see Section 2.1.5) and in a relatively-permanent harm to users' attention, which decreases the effectiveness of all security dialogs and indicators.

From a user's standpoint, security is rarely a primary goal [95]. In the worst case, each presented dialog is simply an interruption and represents a waste of time and an annoyance [6]. In the best case, each dialog represents a small amount of time spent trying to understand both the cause of the interruption and the most adequate answer to this dialog. Based on the results shown in Chapter 7, each presented dialog that is similar to another already-seen dialog with a slightly different message, decreases the response time to subsequent dialogs (see Table 7.3 and Figure 7.5), and increases the chance that a different message will be missed (see Figure 7.4). Thus, increasing habituation increases the probability that users will make an unsafe decision in a suspicious scenario. This, in turn, increases the expected cost of harms.

A variation of the methodology used in Chapter 7 could be used to estimate this cost at the individual level. For example, in the case of the control dialog in Figure 7.5, after 20 exposures of 'irrelevant' dialogs 50% of participants took between 0.85 and 1.99 seconds, with a median response time of 1.2 seconds. Even if we did not know how the response time decreased during those 20 exposures, an upper bound for the cost of responding to dialogs after 20 exposures in the previous example is $\frac{3}{3600} \cdot 20 \cdot w$, where w is a measure of wage per hour. If we take w to be about \$6.5 per hour, then after 20 exposures a user would have lost about \$0.11 in responding to these dialogs. This cost is small compared to the cost of fixing a problem after the user exposed herself to harm. For example, if a user is responding to installation dialogs that are similar to those used in the experiments in Chapter 6, then the harm might be 'infecting her computer with malware', and the cost of 'fixing the problem' is how much it takes to clean up the computer. Even if this time is short (e.g., 20 minutes), it will dominate the previous cost ($\frac{20}{60} \cdot 6.5$ = \$2.1).

The optimal number of dialogs that a person sees in a fixed period of time is such that it does not impose a higher cost than that of fixing the harm that those dialogs are supposed to protect from. If the cost of responding to a number of dialogs is higher than the cost of fixing the harm, then a person would be simply better off by dismissing all dialogs and fixing the harm. Let us

imagine a person that sees n dialogs before exposing herself to a risk on the $n + 1$ occasion. In responding to n dialogs a person loses approximately mn hours, where m is the average time lost per dialog. In most cases, the person will miss an important message in the $n + 1$ dialog, and will lose T hours fixing the problem (e.g., cleaning up her computer after it got infected with malware). If $mn <= T$, then a person should respond to dialogs; if $mn > T$, then a person should simply dismiss all dialogs and fix the problem when it occurs.

Although today we do not have any empirical data on how many dialogs a regular computer user sees on a fixed period of time, this number should include all similar dialogs produced by all applications that a person use regularly. Given the vendor incentives to use dialogs described in the previous sections, it is not hard to imagine a very large number of dialogs being seen by users.

From a vendor standpoint, once a software program is built and distributed to many users there is no cost implied in either the number of dialogs that users have to respond to, or in users having to fix the problems that come afterward. Vendors do not have to assume directly the consequences of unusable software; although sometimes they have to assume the cost of unsafe software [90]. Given that the cost of implementing security dialogs is probably negligible compared to the cost of building the entire software application, the real problem for vendors is not the implementation of dialogs, but what to put in dialogs. Thus, standards and scientific evidence about what is and is not a good dialog may actually solve a problem for vendors.

A number of guidelines have been proposed in warnings research [98], usable security research [33], and by vendors [71] as an effort to avoid the most common errors in security dialog design. For example, Microsoft proposed the NEAT acronym (Necessary, Explained, Actionable, Tested) as a mnemonic to help their engineers remember the most important guidelines to follow when designing computer security dialogs. The original set of guidelines included 68 recommendations described over 24 pages [71]. Although these and other guidelines are valuable contributions, it is not clear how to prioritize among different recommendations, combine possibly conflicting guidelines, or apply them to an existing dialog. Similarly, once a set of guidelines has been applied to a security dialog, it is difficult to isolate and measure the contribution of a single guideline to the improvement of the dialog, thus making it difficult to scientifically validate guidelines effectiveness.

9.3 Policy measures

The two primary problems that this thesis attempts to shed light on are the lack of attention to and habituation to security dialogs. From the point of view of a policy maker, it is important both to encourage software vendors to decrease the total number of security dialogs presented to computer users, and to increase the effectiveness of security dialogs. The following measures are suggested measures that a policy maker could take to meet these two goals.

To encourage vendors to decrease the total number of security dialogs shown to users:

1. **Public perception of dialogs:** A brand-independent campaign against the usage of security dialogs may help to raise public awareness about the excessive number of dialogs. The purpose of the campaign would be to encourage public opposition to the use of security dialogs in software, forcing software vendors to decrease their amount. Such a campaign could be launched even in alliance with the biggest software vendors.

Although I did not find a strong disfavor against security dialogs in my first experiments (Chapters 3 and 4), it is clear from existing literature that computer users dislike both security dialogs and how often these appear (see Chapter 2). One example from popular media is a commercial created by Apple on February 2007 to mock the large amount of dialogs that Microsoft Windows Vista showed to users [4].

2. **CERT-curated, public-fed list of security dialogs:** I suggest creating a list of security dialogs presented to users, curated by a federally-funded organization like CERT or NIST. This list would contain screenshots (or pictures taken with smartphones of laptop or computer screens) showing security dialogs, along with short comments about how the security dialog popped up. Posts would be reviewed before their publication, and the main responsibilities of the curating organization would be to group instances of the same dialog, and to review and allow posts before their publication.

   Having a list of security dialogs displayed by popular software would bring several benefits. It would make clear which software programs display the largest amount of dialogs, allowing to estimate how often these dialogs are shown. It would also give transparency to software purchasing decisions, in a similar way to what happens in expert forums. Given that it is unlikely that software vendors will publish a list of the security dialogs in their software, a list of public-fed security dialogs should be designed and encouraged.

To encourage vendors to design more effective security dialogs:

1. **Federally-funded, university-run security dialog testing program:** The idea is to create one or more federally-funded mechanisms to encourage systematic usability testing of security dialogs by research universities. These mechanisms should ideally be implemented in coordination with software vendors, but vendor participation is not a requirement. The conditions for these grants should include publishing the data obtained in some standardized format to make it available to the public, without excluding the possibility of the publication of academic papers. Alliances between software vendors and university programs should be given special preference over university-only teams. Even if the evaluation is performed without the involvement of software vendors, the publication of the resulting reports would create a public push toward improving security dialogs.

2. **Scientifically tested guidelines in usability and software engineering textbook:** The proposal is to encourage the publication of recommendations like the ones contained in Section 10.2 in every usability and software engineering textbook, and to include them in the contents of every advanced undergraduate and every graduate course on usability and security. This would create a force that would end up, in the long run, in the design of better security dialogs.

3. **Income tax cuts that equal software vendor investments in dialog usability:** Vendors may demonstrably invest in improving their security dialogs, in partnership with universities or research centers, under conditions to be studied (conditions that may include for example a cap on the total amount), and receive an income tax cut for the same amount.

[4] http://www.youtube.com/watch?v=DZSBWbnmGrE

This should create an incentive for vendors to improve their software with some scientific grounding. This incentive should be especially attractive to companies with revenues over $18.3 million[5], which as of February 2014 are taxed a 35% of their income in excess of that amount [43].

All of these suggestions are not mutually exclusive, and may be implemented in a coordinated effort to attempt to solve the two aforementioned problems.

[5]For example, Google, whose total consolidated revenues for 2013 were $59.8 million [39].

Chapter 10

Conclusion

In this thesis, I show that it is possible to study computer users' responses to security dialogs outside of the laboratory when performing realistic tasks, and to do this while keeping the most important benefits of a lab setup (careful control of both the observation process and the level of risk to which participants are exposed), and gaining some of the benefits of online studies (collecting many observations in a cost-effective manner). I also show in this thesis that it is possible to increase the salience of security information in a computer dialog, in a manner that is not eroded by high-habituation conditions.

Throughout this thesis I presented the results of one lab study and nine online experiments. The lab study (Chapter 3) and the first online experiment (Chapter 4) were both exploratory, aimed at understanding how users think about situations in which they would usually find security dialogs. The mental model in Figure 3.1 encapsulates a great deal of knowledge about how users face computer security scenarios.

Further, the experiments described in Chapter 5 allowed me to confirm that it is possible to study the majority of computer user responses to security dialogs with all the properties described above. By focusing on the nuances and difficulties implied by the trusted path problem (described in Section 2.2.2), the experiments also allowed me to explore for the first time the process of attention to security dialogs.

Even further, I conducted another two experiments, described in Chapter 6; I learned from these experiments that it was possible to increase participants' attention to security dialogs. This knowledge would be incomplete without the results of still another two experiments, presented in Chapter 7, where I learned that it was possible to keep participants' attention even after heavy habituation conditions.

Finally, I conducted another two experiments, described in Chapter 8, that allowed me to learn how users respond to more text in a dialog, and to the perception that their antivirus software may or may not be working.

All these studies helped to shed new light on the process of attention and habituation. In the next section, I summarize my findings.

10.1 Findings

10.1.1 Attention and habituation

The problem of how to draw and maintain computer users' attention to information in security dialogs is daunting. Since Dhamija et al.'s seminal work [27], many other authors have studied the problem of attention, mainly to security dialogs [2, 6, 7, 11, 15, 14, 18, 34, 51, 60, 81], and SSL and Extended Validation certificates [10, 45, 83, 84, 87, 94]. However, only a few authors have proposed interface modification techniques to drive computer users' attention to security dialogs [18, 49, 60] (see Section 2.2.3 for a description of their work.)

In Chapters 6 and 7, my co-authors and I proposed and tested a number of different attractors (see Figure 6.10). Each type of attractor was designed based on different ideas of what may increase user attention to dialogs. The ANSI attractor (Figure 6.7) would test whether changing the colors of the text and background of the salient field text was enough to increase attention. The inhibitive attractors were designed to introduce a pause in a user's workflow. Some inhibitive attractors, the so-called forced-action attractors in Figure 6.10, were designed to force the user to complete an action that required a small cognitive effort. Finally, in Chapter 8 I tested several factors in dialogs without any attractors applied.

The following list summarizes the findings in prior chapters:

F1. **Already-habituated participants:** In general, I estimate that between a half and two thirds of participants came to our experiments with an already-existing level of habituation, evidenced by their responses to questions about the content of the dialog and by their response time to dialogs (Chapter 8).

F2. **Novelty effect:** One of the theories that helps to explain computer user behavior is scripts-based understanding (see Section 2.1.5). According to this theory, when a person is making a security decision, she invokes an existing, previously formed script to respond to the situation. Before a script can be invoked, the stimuli in the environment must be scanned, encoded, and compared to the descriptions in the script. If a stimulus is similar enough to what has been encoded, the script is invoked. When a security dialog is different from what users have encoded in their scripts, the person will hesitate, and the environment will be scanned for more stimuli to formulate a new course of action. Based on this theory, I pose that visual attention to an unrecognized security dialog increases. This is what I call Novelty Effect.

    I observed evidence of the existence of the Novelty Effect in the experiments described in Chapters 6, 7, and 8. Simple modifications to dialogs, such as bright and contrasting colors and animations, attract attention momentarily to a dialog. However, this increased level of attention wears off very quickly after a few exposures to the same dialog (Chapter 7).

F3. **Habituation effect:** In Chapter 7, I found that when a set of dialogs that look all alike is presented repeatedly, response time to all subsequent, similar dialogs decreased steadily, and the proportion of users who missed a different message in a new dialog increased. The conditions under which this set of dialogs was presented were artificial and unlikely to be found in real life; however, the experiments described in Chapter 7 helped us to gain a deeper understanding of how an habituated user behaves when responding to security dialogs.

F4. **Attractors:**

(a) **Inhibitive attractors are effective on first use:** Inhibitive attractors significantly reduced the likelihood that participants would pick a risky option when presented with an unsafe scenario, compared both to non-inhibitive scenarios and the control (Chapter 6). However, not all inhibitive attractors remain effective after heavy, repeated exposure (Chapter 7).

(b) **Forced-action attractors are effective even after repeated use:** Swipe and Type attractors significantly reduced the likelihood that participants would pick a risky option when presented with an unsafe scenario (Chapter 6), and are resilient to habituation (Chapter 7).

(c) **Forced-action attractors impose a usability burden:** Inhibitive, non forced-action attractors delay users slightly more than not using any attractors. Forced-action attractors initially impose a much larger delay on users' response than non forced-action attractors, although this delay decreases with habituation. Out of the forced-action attractors, Swipe imposes the shortest delay on users, both with and without habituation: 75% of users learn to dismiss dialogs using the swipe attractor in less than 5 seconds under high-habituation conditions. The Type attractor is especially effective under high-habituation conditions (Chapter 7).

F5. **Dialog content:**

(a) **Shorter dialog texts are better:** In Chapter 8, I found that when text that is not related to security is added to a security dialog (up to a point), the proportion of participants who did not pay attention to the salient field in a suspicious scenario increases.

(b) **Earlier is better:** In Chapter 8, I found that security advice placed at the end of a security dialog in a suspicious scenario decreases the proportion of participants who paid attention to the salient field.

10.1.2 Comprehension, expertise, and demographics

The lab study described in Chapter 3 and the online experiment described in Chapter 4 allowed me to delve a bit into how knowledge and other variables affect human response to security dialogs.

As I discussed in Section 2.2.1, a number of authors have studied how comprehension affects user response to security dialogs [34, 64, 87]; however, comprehension is an overloaded concept, and authors have defined it in different ways depending mainly on their own experimental setup. In the experiment described in Chapter 4, I defined comprehension as "understanding of the problem that triggered the security dialog", that is, being able to recognize and identify the problem. Note that this is different from knowing how to solve the problem, or the consequences of the problem. In the experiment, my co-authors and I found that being able to identify the problem was mostly uncorrelated with safe response.

Experienced and novice users observe different cues, and arrive at different conclusions when they are presented with a security dialog. Experienced users take preemptive measures against problems, while novice users tend to react to problems. The language that experienced and novice

users understand is different. Consequently, it is possible that experts and novice users need different types and amounts of information in security dialogs to make the best use of dialogs. While experienced users need to know about the specific problem (e.g., this is a self-signed SSL certificate), novice users need to be reminded of the practical consequences of a problem *without* any technical jargon (e.g., your connection to this website is protected, but the website operators may not be who you think they are). This notion suggests that dialogs should adapt themselves to users' level of expertise. This idea has already been suggested by Keukelaere et al. [49].

Eight out of nine online experiments included a very complete set of demographic questions: gender, age, occupation, ethnicity, education, and knowledge of at least one programming language. I did not find consistent evidence that any of the studied demographic variables was significantly correlated with any outcome variable (e.g., understanding of the problem presented by a dialog, safe response, perception of importance of the problem presented by a dialog [*motivation*], having anti-virus installed, care for the computer that the participant was using, and others.) Although the Human-In-The-Loop model includes demographic variables as factors that possibly affect human response to security dialogs, I did not observe any consistent evidence supporting this. Furthermore, evidence of demographic-based differences in response to security dialogs was scarce in the reviewed literature; one exception was the work of Krol et al., in which participants who complied with a dialog in a lab experiment were "overwhelmingly female" [51].

10.2 Recommendations

Although my co-authors and I designed and tested an effective technique to increase user attention in a way that is resilient to habituation, I strongly discourage the utilization of this technique for all dialogs in a computer system: attractors should be applied only to those dialogs which present users with decisions having severe, potentially irreversible consequences.

The recommendations below attempt to match the previous findings, and are directed to web designers and software programmers in charge of a) deciding when to implement a security dialog, and b) when showing a dialog, deciding what to put into the dialog.

R1. **Do not implement a dialog unless it is absolutely necessary.** Given that users are already habituated to dialogs, it is imperative to avoid showing a dialog unless it is absolutely necessary. If a dialog is shown, every effort must be done to drive user attention to the important security information in the dialog.

R2. **When presenting a dialog, make it consistent and predictable.** Although a change in the appearance of a dialog will attract attention, this increase in attention will wear off quickly. Every time a dialog calls for user attention unnecessarily, it aggravates existing user habituation. Security dialogs should be presented consistently and predictably, and user attention should be called for only in those cases where it is deemed strictly necessary.

R3. **Before implementing a dialog, estimate its frequency of appearance and the degree of harm it will protect the user from:**

   (a) If a dialog is expected to be shown only once or very rarely, then use a technique that is effective on first use (i.e., an inhibitive attractor).

(b) If a dialog will be shown many times and the consequences of a risky action are not severe, use a technique that is effective even after repeated use and that it does not impose a great usability burden on the user (i.e., a forced-action attractor like Swipe).

(c) If the dialog will be shown often and the consequences of a risky action are severe, use a technique that is effective even after repeated use, and that imposes a relatively large usability burden (i.e., a forced-action attractor like Type).

R4. **Make your dialogs concise; put the important advice first.** When designing the content of a dialog, make your message as short as possible. Put the important security advice first.

R5. **When designing an interface technique to call for user attention, design a mechanism that:**

(a) Forces foveation over the salient field – i.e., forcing the user to look directly at the salient field.

(b) Requires a small amount of cognitive effort to enable the risky option.

(c) Is activated only when the user signals her intention to pick a risky option.

(d) Prevents the user from immediately picking the risky option.

Although I show in Chapters 6 and 7 that attractors are effective, other techniques to increase user attention can be created. This is the rationale behind the last recommendation in the list above. Some attractors were effective because they required participants to perform a task that forced foveation over the information we wanted to drive attention to (the salient field). Part of the success of attractors can be attributed to tasks that require some level of cognitive effort (e.g., typing a text). In the case of the Swipe attractor, participants were required to move the mouse from left to right over a relatively thin strip; our participants reported later that the task was not an easy one. In the case of Type, it required users to read and retype a sentence. Request, one of the inhibitive attractors that did not work (see Figure 6.6), required only to click on a small, secondary pop-up; it follows that not every simple user action produces the desired effect, but only those actions that require some (small) cognitive effort. Apparently, for the user who knows how to use a mouse or a touch-pad, clicking on an option required no cognitive effort at all. The first recommendation in R5 (R5.a) is necessary because we need users to drive their attention to the salient field. Recommendations R5.b through R5.d are necessary to adequately balance usability (R5.c) and security (R5.b and R5.d). To illustrate, a hypothetical security indicator that:

1. Did not require a small cognitive effort (i.e., omitting R5.b) would fail to prevent a user from getting habituated. Our Reveal attractor (Figure 6.3) is an example of such a mechanism.

2. Required to confirm every action by (for example) typing a sentence (i.e., omitting R5.c) would be unacceptable from the point of view of usability.

3. Allowed to pick the risky option immediately (i.e., R5.d) would fail to stop a user that is already habituated. Our unqualified Animated Connector attractor (Figure 6.2) is an example of such mechanism.

10.3 Future work

A question that should be explored further is what criteria should be used to decide when showing a dialog is of the utmost importance, so a system can decide when to show it. Methods used to date to counteract threats such as phishing are a mixture of manual (e.g., phishing website black-lists) and automatic (e.g., bayesian email filters) mechanisms.

I proposed and tested a technique to increase user attention to salient fields in security dialogs. However, I strongly believe that this technique should not be used for every single dialog shown to the user. Instead, we should use it as a last resort, from a user interaction design standpoint. A critical question to answer is: when is it appropriate to show a security dialog with an attractor applied to it? Any decision mechanism will be programmed, so it needs to be completely automatic. It seems natural that such a decision takes user behavior as an input. Can we create an automatic mechanism (i.e., an engine) that learns from users' response time (for example), and decides when it is appropriate to use an attractor? In the case that such a mechanism can be built, is it more efficient than a) showing only attractors and b) showing no attractors? An experiment could be conducted by developing an extension for a browser, or by modifying an already existing application for Android. We would invite participants to install it and evaluate it for a while. In a first stage, we would collect user information (e.g., timing data) to tune an engine that could modify dialogs when required. In a second stage, we would let the engine 'decide' how to intervene in the dialogs, and then collect participants' responses to dialogs to evaluate whether such an engine is effective.

Another important question is whether training users to recognize malware is an effective way to decrease the impact of malware, in the same manner in which Kumaraguru et al. used especially crafted email messages to train users to distinguish phishing messages [52]. It is important to explore the creation of embedded training to help users to pay attention to and recognize those signals that indicate the likely presence of malware, for example, a signed application wherein the certificate has been self-signed.

I envision an experiment in which users are directed to training material about malware *right after* they clicked on the 'Install' option in a security dialog – making use of a 'teachable moment.' I would use a similar method to the one described in Chapter 5 to simulate dialogs. The experiment should be split in two parts. In the first part, users may be invited to evaluate website apps, like KanbanFlow (`www.kanbanflow.com`) or ToDo.ly (`todo.ly`); the last of a small number of applications would be a simulated one, which requires a new plugin to be installed. Those participants that choose to install the simulated plug-in would be taken to a page where they are taught how to identify fake applications. In the second part of the experiment, the same participants would be invited to take part "in a another experiment, run by different researchers," in which the presented primary task would be different but the scheme is the same: users are asked to install an application. Then, a repeated-measures comparison is made to understand whether the proportion of participants who installed the second time is lower than the first time.

Appendix A

Expert mental model

Description

The mental model presented here follows the usual conventions of mental models, as presented by Morgan et al. [63]. The model is an influence diagram; the arrows show some degree of causal relation. Ovals are chance nodes, corresponding to actions that may or may not be taken by an expert user, or events that may or may not happen. Rectangles are decision nodes: actions that can be taken purposefully to change the likelihood of something to happen.

The built expert mental model summarizes the information obtained from the coding and analysis of 10 interviews with experts, and follows in general terms the "meta-model" shown in Figure A.1. The following concepts were found in all experts transcripts:

Variables: Events or conditions that an expert considers while understanding and evaluating a risky situation. A variable may lead an expert to think into another variable. All variables in the expert model are shown in Figure A.2.

Outcomes: Situations that may affect negatively the user of a system. All outcomes are shown in Figure A.3.

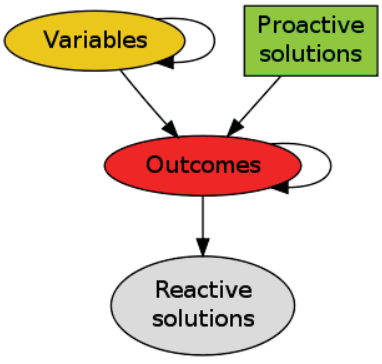


Figure A.1: General aggregated model developed from interviews with experts.

Proactive solutions: Actions that experts perform to prevent a problem from appearing. Usually the knowledge of these actions come from the expert's experience. All proactive solutions are shown in Figure A.4a.

Reactive solutions or "reactions": Actions applied to solve a problem that has already happened. All reactive solutions are shown in Figure A.4b.

The full expert model (shown in Figure 3.1) does not show all possible connections between concepts but only those that were considered as most important. An arrow in the model shows a relation between two concepts, as perceived and mentioned by the experts.

Variables

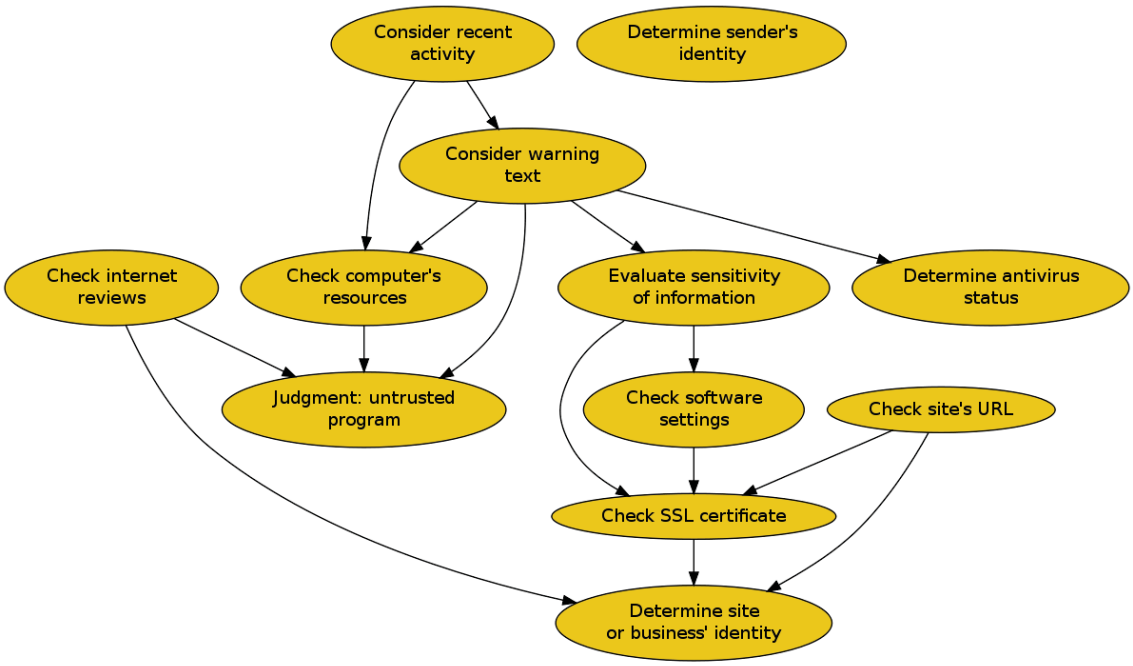


Figure A.2: Variables in the expert mental model. Arrows show recurrent relationships between variables.

Check computer's resources: This node represents any action towards finding out how many resources are currently available in the computer: RAM, available disk space, number of processes being run, availability of Internet connection, etc. It may involve many operations or a single look to a software showing this kind of information. Different operating systems may offer more direct ways of obtaining this information. For example, most Windows managers for the Linux OS include a "system monitor" that can be configured to permanently show on the screen the amount of resources currently being used by the computer.

Check Internet reviews: Most experts use information published on technical forums, mailing lists or other experts' evaluations to understand more about a problem or risk. This node represents the action of looking for any specific information in such sources.

Check site's URL: Since URLs are an essential part of navigating through Internet, experts often check URLs of resources in their browsers before using them. This might be done by checking several different places (location bar, status bar, certificates, etc.)

Check software settings: Most experts understand how to configure the software they use, and usually tweak that configuration to meet their changing requirements. This node represents the action of checking and eventually changing software settings as a consequence of other actions being performed before, or to obtain a specific effect.

Check SSL certificate: SSL certificates are used in many different contexts. This node represents the action of checking an SSL certificate before involving oneself into a transaction.

Consider recent activity: When faced with a specific situation, all experts wonder immediately about the actions being performed right before the warning appeared. The variety of actions being considered is broad, and includes anything being done from seconds to minutes before the considered moment. Experts usually inquire about those actions being performed, and decide which of these activities are relevant for the problem and which are not. This node represents the expert action of inquiring for those activities recently performed in the system.

Consider warning text: Experts usually read carefully the warnings they are presented. This node represents this reading.

Determine antivirus status: Often viruses are a direct or indirect cause for the problems observed in a system. Experts usually inquire whether the operating system is architecturally safe (Linux and BSD-based, such as Mac OS); in case it is not, whether there is a working antivirus installed in the system, and whether it is up to date. This node represents such inquiry.

Determine sender's identity: Many interactions carried over the Internet involve determining the identity of the sender of certain piece of information, such as an email, a file or a website. Usually this consideration can be phrased as "Do I know this person/site/file?". It might include looking for other emails of the same person, or looking for the site within a personal bookmark. This node represents those actions performed towards obtaining more certainty about such identity.

Determine site or business' identity: This node is analogous to the node related to "sender's identity". The process of determining the identity of a website is different though, and it has different problems associated to it.

Evaluate sensitivity of information: Experts often evaluate the sensitivity of the information being sent or received against the effort of sending it encrypted or not. This applies both to emails and to web pages with forms.

Judgment: untrusted program: This node represents a judgment call about a software. Experts judge might consider a software as unreliable because they consider it malicious or buggy, and advise against executing it or even having it installed.

Outcomes

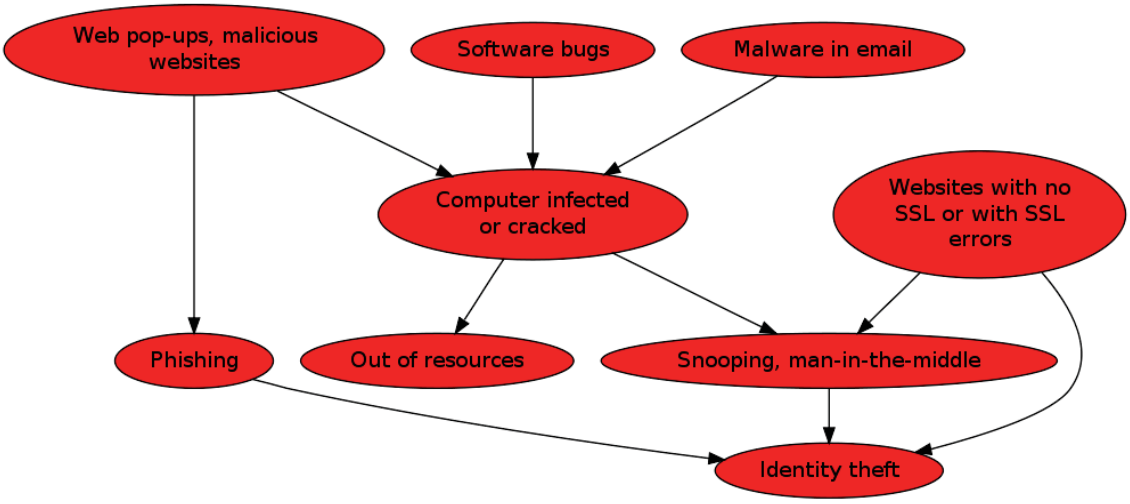


Figure A.3: Outcomes in the expert mental model. Arrows show recurrent relationships between outcomes.

Computer infected or cracked: One of the most commons problems is having one's computer infected by viruses, trojans or any other kind of malware, or even cracked remotely by someone. This node represents that situation.

Identity theft: This node represents the problem of "identity theft", or more precisely, of impersonation. It is an external problem, in the sense that once identity data has been stolen, there is no way of reacting to the problem in terms of the system itself (all remediation measures are outside of the scope of the system).

Malware in email: This node represents the problem of infecting one's computer by opening email with malicious content, such as HTML with malicious ActiveX content in the MS Windows OS.

Out of resources: This node represents the exhaustion of resources: RAM, hard drive, space, processor time, network connection bandwidth, etc.

Phishing: This node represents the problem of phishing, including fake emails and spoofed websites that mimic others, designed to trick a person into disclosing private or sensitive data.

Snooping, man-in-the-middle: It represents the problem of being eavesdropped while sending or receiving information. This situation applies to emails, websites and transactions over the Web.

Software bugs: Some software might not be malicious per se, but it might contain bugs that make the software risky to use. This node represents the problem of installing and using such a software.

Web pop-ups, malicious websites: A common problem is related to websites that have malicious scripts, often aiming to force the user to see an Ad, or accepting a situation that is not desired. This node represents such a situation.

Websites with no SSL or with SSL errors: This node represents websites which are supposed to secure the connection with users but do not have an SSL certificate, or if they do, it has been incorrectly configured.

Proactive solutions

Don't open random files with certain programs: Some files can have deleterious effects either on a person's computer or data when "activated" (that is, double clicked or opened through a specific software). This node represents the advice of not opening files whose origin or purpose is uncertain, especially with certain known dangerous software (e.g., .doc files with MS Word).

Minimize exposure to Internet, use safer software: This node represents the strategy of minimizing exposure to Internet of a computer to protect the information stored on it. This is to prevent infection by viruses or malware, or to avoid spreading an infection when the computer has been already infected. It is also applied to minimize deletion, corruption or disclosure of stored data.

Obtaining URLs safely: This node represents the obtainment of "safe" or "trusted" URLs, either by typing them in or by recover them from a personal bookmarks.

Prefer established brands: This node represents the preference for established and well-known brands for buying products or hiring services through Internet.

Take defensive actions on email: This node corresponds to three different actions performed on email: not opening links contained in emails, not opening unexpected emails, and trusting only messages from known people.

Use secure connection tools: This node represents the usage of different tools to secure a connection to a website or an email or application server, which typically includes an SSH connection.

Reactive solutions

Defensive measures in computer: This node represents five specific actions taken on a computer:

1. Hard reboot or turn off the computer.

2. Quarantine the computer by plugging-off or turning off the network connection.
3. Running an antivirus program check over hard-drive files.
4. Reinstall the operating system, or deleting the whole content of the hard-drive.
5. Upload all suspicious files to VirusTotal2.

Don't allow access if unsure: This is a conservative strategy aimed at "not accepting" or "not allowing" any unknown, uncertain or suspicious request.

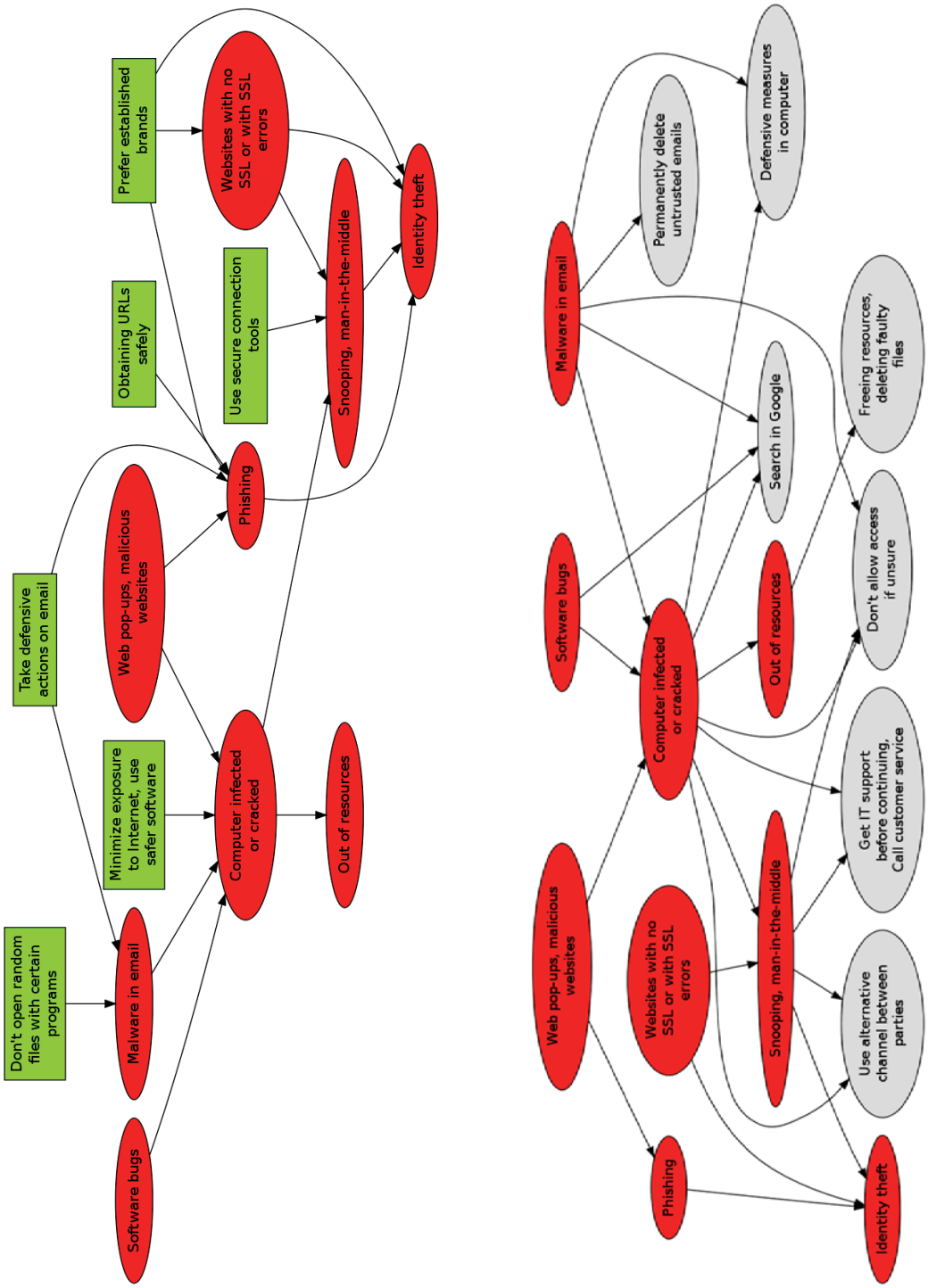
Freeing resources, delete faulty files: This node represents the action of freeing or releasing used resources (memory, hard-drive, processor, etc.) to gain more resources for a specific task.

Get IT support before continuing, call customer service: When the specific condition being faced seems to be not solvable by the person, external help is needed. This node represents such a condition.

Permanently delete untrusted emails: This node represents the action of deleting all unknown or suspicious emails from the personal Inbox. This explicitly means not to keep the email in the Trash folder, since many people do this, which might re-infect a computer once the risk has been overcome.

Search in Google: Often experts search for literal texts in Google to obtain more information about a condition, warning message or error dialog that they don't know how to handle. This node represents the action of taking part of a message and looking for it in Google (or any other search engine).

Use alternative channel between parties: Whenever there is a connection between two parties that might have been compromised, experts often advise to use an alternative channel (even not digital channels) to talk the intended recipient of the communication. This node represents such action.



(a) Proactive solutions (in green) and outcomes (in red).

(b) Outcomes (in red) and reactive solutions (in gray).

Figure A.4: Proactive solutions, outcomes, and reactive solutions in the expert mental model.

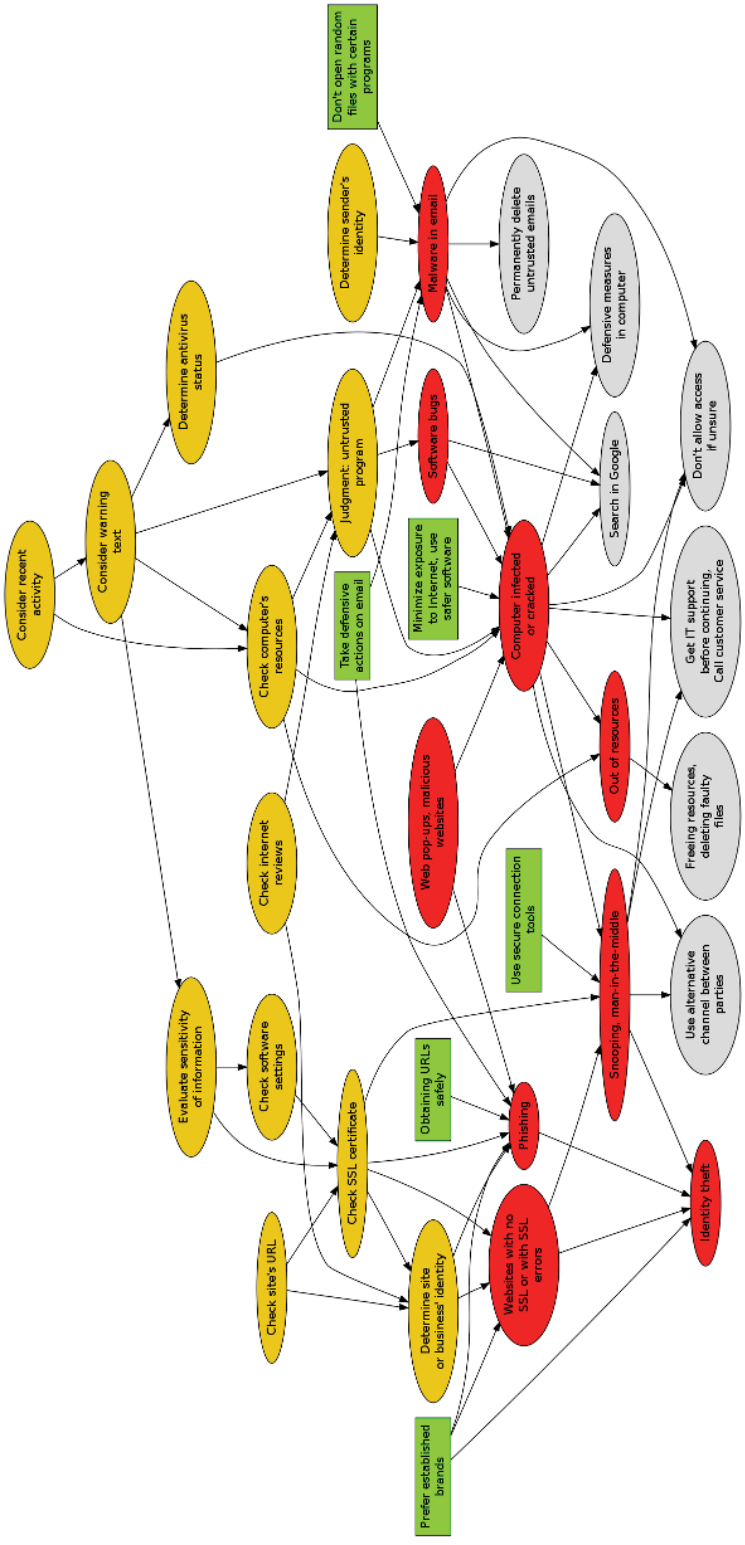


Figure A.5: Full expert mental model.

Appendix B

Interview script for mental model study

Greeting

Thank you for being here today.

This interview is part of research at Carnegie Mellon, aiming to understanding what activities people value being able to perform on their computer, either at their home, their job, or their places of study. Also, to understand what people consider a risk or threat when they are using a computer system and to understand how people react when they encounter a computer warning in several everyday scenarios.

This interview will be anonymous and confidential. It will take us approximately an hour, and will be divided into three parts. In the first one, I will ask you some general computer related questions. In the second one, I will describe a fictitious scenario and ask you what would you do in such a situation. Then in the third, I will ask some general questions. I would like to stress the fact that we are not testing you in any way. Please feel free to say anything that may come up to you during the interview.

Before we start, do you have any questions?

[Begin audio recorder.]

Open questions about activities and risks

Please think about all computers available to you, either at home, at your work or place of study, or those that you carry with you (iPhone, Blackberry, etc.), or even those computers that do not belong to you, but that you use anyway (for example, your husband/wife's computer, your friends' computers, etc.)

What activities or tasks do you value being able to perform with these computers? (Prompt: sending emails?)

Anything else? (Keep asking, "Anything else?" until they say No.) (Make list of activities/tasks they mention.)

What kind of problems do you think you might encounter when doing [this activity] [use list from previous question]? (Prompt: What kinds of things might happen to your computer by doing [an activity they mentioned].) (Ask for each activity mentioned.) (Make list of problems they mention.)

Why do you think [this problem] might happen? What measures can you take to avoid/fix [this problem]? (Prompt: A problem might occur because of the reliability of software?) (Use list of problems they mentioned, ask for each problem.)

Scenarios description

Description only for experts: Imagine you have a close friend, Paul, who knows nothing about computers. He knows you are an expert, he trusts you, and he has asked you for help in case he runs into problems using his new laptop. You have agreed to help him. One day, he calls you on the phone, and you can notice by his voice that he is very nervous. He tells you the following.

Description only for non-experts: Imagine you have a close friend, Paul, who knows nothing about computers. He thinks you are a more experienced computer user than him, he trusts you and he has asked you for help in case he runs into problems using his new laptop. You have agreed to help him. One day, he calls you on the phone, and you can tell by his voice that he is very nervous. He tells you the following.

Scenario 1: Disk-full warning (1: Information deletion or loss)

I was reading my email, and I remember I saw Rob the other day using this awesome screensaver of a waterfall, with authentic sound and everything, so I thought of looking for one of those. It took me a while but I found it, and the website said that I should download it and click on it twice. So I did that and everything was fine. I started to see some pictures of my daughter, and then this happened. (Show warning #1) I don't know what's wrong, it was working all right.

1. Could you please tell me what this message is?
2. (Skip this question if participant says "I dont know" to the previous) What does that mean?
3. What do you think will happen if I click on OK?
4. (Skip this question if it is already answered) Is that bad?
5. What should I do?

Scenario 2: Encryption-conflict warning (2: Information disclosure)

(A few days later, Paul calls you again and says) I was trying to send one of these funny postcards to Rob, because today's his birthday. I have never used Outlook before, so I don't know what's going on. The only thing that I did was to type the greeting on a message and to click on the 'Send' button. [Show warning #2]

1. Could you please tell me what this message is?
2. (Skip the following question if participant says "I dont know" to the previous) What does that mean?

3. What do you think will happen if I click on ‘Send unencrypted’?

4. (Skip this question if it is already answered) Is that bad?

5. What should I do?

(Next week he calls again, very concerned and tells you) I’m sorry to bother you again, but it seems that I have the same problem as last week, remember? But now I’m trying to send my boss some information, he asked me about a possible fraud occurring here at the office, a real big problem for the company and for some employees.

1. What do you think will happen if I click on ‘Send unencrypted’ this time?

2. What should I do?

Scenario 3: Address book warning (4: Execution of malicious code)

(An hour later Paul calls you again) I’m terribly sorry, but a new message has just showed on my screen. I’m a nervous wreck. (Show warning #3)

1. Could you please tell me what this message is?

2. (Skip the following question if participant says “I dont know” to the previous) What does that mean?

3. What do you think will happen if I click on ‘Yes’?

4. (Skip this question if it is already answered) Is that bad?

5. What should I do?

(In response to any inquiry about how the message appeared or what his actions were leading up to the message appearing.) Well, I was reading my email, and a message appeared from my bank, and a message from Carol, my wife, and also a message from Rob inviting me to his Birthday. After that a message appeared saying that I had a virus or something like that, but I clicked on ‘Clean it’ and the message went away. And now I’m stuck with this message, and I don’t know what to do.

Scenario 4: Attachment warning (4: Execution of malicious code)

(Imagine you are a bit bored of this friend, and you send him by email a digital copy you bought of “Computers for dummies.” After a few hours, he calls you up again.) Thank you very much for the book! I will try to read it, so I dont bother you as much, but... well, I’m sorry, but it happens that when I clicked on the book something happened again. (Show warning #4)

1. Could you please tell me what this message is?

2. (Skip this question if participant says “I dont know” to the previous) What does that mean?

3. What do you think will happen if I click on ‘Open’?

4. (Skip this question if it is already answered) Is that bad?

5. What should I do?

You know what? This same thing happened to me a couple of weeks ago, with one of these funny chain emails... I tried to click on the email and then this appeared on my screen, and it annoyed me a bit so I just tried to get rid of it and I clicked on the email again, and the same thing happened.

1. What do you think would have happened if I clicked on ‘Open’?

2. What was I supposed to do?

Scenario 5: Unknown certificate warning (5: Trust in malicious third-parties)

(After a few weeks of not calling you) Hi, I’m sorry to bother you again. I’ve been reading the book and it’s quite interesting. I’ve solved most of the problems I’ve seen myself, but I don’t know what to do with a very confusing message that has just appeared on my screen. I was visiting a website, and suddenly this message appeared. (Show warning #5)

1. Could you please tell me what this message is?

2. (Skip this question if participant says “I dont know” to the previous) What does that mean?

3. What do you think will happen if I click on ‘Yes?

4. Is that bad?

5. What should I do?

(If inquired about what kind of website) Well, it’s the website of a small online store where I bought some toys for my daughter. I would like to give her a present for her birthday. I bought something here about a year ago, but it was on a different computer. I don’t remember having seen something like this.

The other day I was trying to check my online bank account and the same message appeared. Is that any different?

1. What do you think would have happened if I clicked on ‘Yes’?

2. Is that bad?

3. What was I supposed to do?

General ending questions

1. Have you ever received any of these five warnings?
2. Have you seen any other types of warnings?
   (a) Was it ever unclear what a warning meant? [Prompt: Do you remember anything that was unclear?]
   (b) How do you usually respond to these warnings? [Prompt: Do you close them, or click 'OK' or look for more information?]
3. (Pull out their list of problems they stated they might encounter while using their computer) (Ask for each proble) Have you ever seen any type of warnings when you encounter (this problem)? What type of warnings? Was it ever unclear what that warning meant?
4. Would you do the same that you advised this fictitious friend to do?
5. What kind of computers do you use? What kind of operating system do these computers have?
6. What program do you use to read your e-mail? Do you use any other email programs?
7. Do you have a spam filter on your email account?
8. What web browser do you usually use to access the Internet? Any other?
9. Do you have any type of software or programs on your computer that is supposed to help keep your computer safe?
   (a) What are they?
   (b) What does that program do?
   (c) (If not mentioned) Do you have antivirus software installed on your computer? What software do you use?
10. What is your occupation? How old are you? (write the gender of the person)

Appendix C

Experimental material in credentials study

C.1 Participant solicitation for credentials experiment

Researchers at Carnegie Mellon University are conducting a set of brief surveys about online games. You will have to play three online games, and then answer a short survey giving us your opinion about each game. The whole survey should take you about 20 minutes. We will pay you $1.00 for your participation.
Requisites to participate:

1. You must be 18 years old or older.
2. You must be in the United States while you take the survey.

3.w [*shown only to users Windows clients*]
You must use Microsoft Windows Vista or Windows 7. We will not pay you if you use another operating system, or an older version of Microsoft Windows (like Windows XP). You don't have to use MS Internet Explorer, but if you do you must use Internet Explorer 8 or higher.

3.m [*shown only to users of MacOS clients*]
You must use Apple MacOS to participate. We will not pay you if you use another operating system.

3. You cannot take this survey twice. Please click here to check if you have taken this survey before.

To be paid, follow these steps:

1. Go to: `http://saucers.cups.cs.cmu.edu/yacot/mnt/wtk/survey.php?i=workerID`
2. After completing the survey you will receive a confirmation code in the last page. Enter the code in the box below and we will approve your payment. Please do not enter the code more than once. If you are not sure about having entered the code correctly, please send us a message and we will solve the problem as soon as possible.

C.2 Example game evaluation form

Instructions to evaluate the game:

1. While pressing CTRL/Command on your keyboard click on the link below to open the game in a new tab of your browser.

2. Click on the button ”Click to play online” on the left of your screen. Wait for the game to load.

3. When the game has loaded completely, play the game ”Mars Buggy Online” for about 2 to 3 minutes.

4. Return to this survey to answer the questions below.

Assigned game N: Mars Buggy Online http://www.gametop.com/online-free-games/mars-buggy-online/ (Press CTRL/Command while clicking this link) **Attention**: The website whose URL appears above is external to this study. Our researchers **do not** control its content.

If you are not able to download or install the game above, please check the box below and then click ’Next’ on the bottom of the page. You will be assigned a new game to evaluate.
☐ I was not able to download or install the game, please assign me another game to evaluate.
Please tell us briefly why you were not able to play the game: (required open text)

Please enter here a one-sentence description of the game you played (between 10 and 50 words): (optional open text)

Please answer the following questions about the game you played:

Have you ever played this game before?
◯ Yes
◯ No

Do you think this game is appropriate for children between 4 and 8 years old?
◯ Yes
◯ No

Do you think this game is appropriate for pre-teenagers between 9 and 12 years old?
◯ Yes
◯ No

Do you think this game is appropriate for teenagers between 13 and 17 years old?
◯ Yes

◯ No

Do you think this game is fun?
◯ Yes
◯ No

C.3 Exit survey

Note: The logic of the online survey is not shown here. New pages on the online survey are indicated by an horizontal line.

Since the moment you started this study, have you encountered any requests to enter a password, including requests that may have come from a website, your browser, or your computer's operating system?*(mandatory question)*
◯ Yes
◯ No

What program or website requested your password? [Text field]

Below you will find several security windows. If you have seen one of these windows, or a window that is very similar to one of these three, please click on it.
◯ MAC1&2
◯ WIN1
◯ UAC1
◯ wrong1
◯ wrong2

_____ Is the password-entry window that appeared during this study one that you had seen before you participated in this study?*(mandatory question)*
◯ No, I had not seen this window before
◯ I've seen a similar window before, but the window looked different
◯ Yes, I've seen this window before
◯ I'm not sure

_____ You entered l characters into the password field of the dialog box. This password you entered is stored in your browser but we have not sent your password to our servers. Was the password you entered a real password for an account on your computer?*(mandatory question)*
◯ Yes, I did enter a genuine password (we'll immediately delete any records of the password we kept in your browser)
◯ No, I did not enter a genuine password

Since you did not enter a genuine password into the password field of the dialog box, may we collect the contents of this field for analysis?*(mandatory question)*
◯ Yes, since the password I entered is not a genuine password you may send it to your servers.
◯ No, the password I entered was actually a genuine password. Please immediately delete any records of it in my browser and do not send it to your servers for analysis.
◯ No, I have another reason for not wanting the password I entered sent to your servers (please explain).: [text field]

You entered '___' as your username. What do you use this username for?*(mandatory question)*
◯ I use it to enter a personal computer
◯ I use it to enter a work account belonging to a business with 5 or more employees
◯ I use it to enter a work account belonging to a business with 4 employees or fewer
◯ I use it to enter a school or university account
◯ I use it to enter a website
◯ I made it up, I didn't want to enter anything real.
◯ I use it for other purpose (please explain): [text field] *(mandatory question)*

Since you made that username up, can we collect the content of that field for analysis?*(mandatory question)*
◯ Yes, since I made it up, you can keep that username
◯ No, that is an actual username (we will not keep this username)
◯ No, for other reason (please explain): [text field] *(mandatory question)*

_____ Please explain why you didn't enter your password into the password-entry window*(mandatory question)* [textarea]

_____ Please indicate which (if any) of the following factors contributed to your decision of not entering your password*(mandatory question)*
☐ I didn't want to take the time to install the software
☐ I didn't want to put new software on my computer
☐ I was concerned that the software I was being asked to install could slow down or crash my computer
☐ I was concerned that the software I was being asked to install could harm my computer or steal my information.
☐ I didn't understand why I needed to enter my password.
☐ I thought that the password-entry window was trying to steal my password.
☐ Other, please specify

Please describe briefly why you think the password-entry dialog popped up and requested your user name and password.*(mandatory question)* [textarea]

_____

How strongly do you agree or disagree with the following statements?*(mandatory questions, all having the same choices as the one below)*

♦ The password-entry window made me feel confident that MS Windows/MacOs was protecting my online safety
◯ Strongly disagree
◯ Disagree
◯ Neither agree nor disagree
◯ Agree
◯ Strongly agree

♦The password-entry window made me feel anxious that the plug-in I was downloading might be dangerous

♦The password-entry window made me feel confused

♦The password-entry window made me feel annoyed about having to enter my password

How safe are the following activities?*(mandatory question, all having the same choices as the one below)*
♦Installing browser plug-ins
◯ Very unsafe
◯ Unsafe
◯ Neither safe nor unsafe
◯ Safe
◯ Very safe

♦Downloading programs for my computer

♦Playing games in my web browser

♦Downloading games for my computer

♦Taking online surveys

_____ During this study you were asked to install an updated Silverlight plug-in. Clicking "Yes" to install the plug-in would cause the plug-in to be added:*(mandatory question)*
◯ For the current tab/window and browser session
◯ For the current browser session for all tabs/windows
◯ For all tabs/windows for the life of my browser/computer
◯ Other, please specify: [text field] *(mandatory question)*

During this study you were asked to install an updated Quicktime plug-in. Clicking "Yes" to install the plug-in would cause the plug-in to be added:*(mandatory question)*
◯ For the current tab/window and browser session
◯ For the current browser session for all tabs/windows

◯ For all tabs/windows for the life of my browser/computer
◯ Other, please specify: [text field]*(mandatory question)*

Prior to this study, did you have Microsoft Silverlight installed on your computer?*(mandatory question)*
◯ Yes
◯ No
◯ I don't know

Prior to this study, did you have Quicktime installed on your computer?*(mandatory question)*
◯ Yes
◯ No
◯ I don't know

_____ Please do not reveal what we are about to tell you to others who have not yet taken this study, as doing so may change the behaviors we are trying to observe. Most of what you did in this study was real. However, the password-entry window you saw was actually part of the content of the website. We were mimicking the windows from our browser to collect data on how users respond to window that just appear and request a password.

The study is designed to help Carnegie Mellon University and Microsoft understand how to keep people safe online. To do that, we are simulating some of the tricks that malicious websites might use to gain access to your computer. In this case, the track we are simulating is one in which an attacker creates an image that mimics a password-entry window. If you entered your password in the window that appeared, please note that did we not observe, record, or transmit your password.

As researchers, we take the safety of our participants very seriously and we are required to minimize the risk you undertake by participating in the study. We are entirely reliant on your honest response to our survey questions to determine if, had this been a real attack, the attacker would have captured your password. We are depending on you to answer questions honestly and completely. If you have any concerns about this study, please contact the researchers indicated at the end of this survey to ask questions or express concerns.

Did you know that the password-entry window was actually mimicked by the website, and not a real password request from your operating system or browser (Internet Explorer)?*(mandatory question)*
◯ I never suspected
◯ Something felt funny or suspicious, but I had no idea what it was
◯ I suspected that the password-entry window wasn't real. Please explain: [text field]*(mandatory question)*
◯ I was completely sure that the password-entry window wasn't real. Please explain: [text field]*(mandatory question)*

Why did you suspect the password-entry window wasn't real?*(mandatory question)*
☐ I moved my browser window and the password-entry window moved with it
☐ I resized my browser window and the password-entry window was hidden
☐ I tried to move the password-entry window and it did not move
☐ Password-entry windows flash three times when they first appear, whereas this window did not flash
☐ The window's color scheme did not match my browser's color scheme
☐ I was told that the password-entry window would be fake
☐ Other (please explain): [text field]

When you encountered the password-entry window, did you suspect that the purpose of the study might be to examine whether you would provide your username and password?*(mandatory question)*
◯ I never suspected the password-entry window was an important component of the study
◯ I had some suspicion that the study was about the password-entry window, but wasn't sure
◯ I was strongly suspicious or was certain that the study was about the password-entry window
◯ I knew ahead of time that the study was about the password-entry window (please explain how you knew): [text field]*(mandatory question)*

_____ Please describe carefully what you did to play the game at the last website you visited:*(mandatory question)* [textarea]

_____ Does the computer you are using have security or anti-virus software?*(mandatory question)*
◯ Yes
◯ No
◯ I'm not sure

Do you know any programming languages?*(mandatory question)*
◯ Yes (please indicate which programming languages you know): [text field]
◯ No

What is your gender?*(mandatory question)*
◯ Female
◯ Male
◯ Decline to answer

What is your age?*(mandatory question)*
◯ 18
◯ 19
(...)
◯ 98
◯ 99

◯ Decline to answer

What is your race/ethnicity?*(mandatory question)*
◯ Asian/Pacific Islander
◯ Black/African-American
◯ White/Caucasian
◯ Hispanic
◯ Native American/Alaska Native
◯ Other/Multi-Racial
◯ Decline to answer

What is your current occupation?*(mandatory question)*
◯ Administrative Support (eg., secretary, assistant)
◯ Art, Writing and Journalism (eg., author, reporter, sculptor)
◯ Business, Management and Financial (eg., manager, accountant, banker)
◯ Education (eg., teacher, professor)
◯ Legal (eg., lawyer, law clerk)
◯ Medical (eg., doctor, nurse, dentist)
◯ Science, Engineering and IT professional (eg., researcher, programmer, IT consultant)
◯ Service (eg., retail clerks, server)
◯ Skilled Labor (eg., electrician, plumber, carpenter)
◯ Student
◯ Other Professional
◯ Not Currently Working/Currently Unemployed
◯ Retired
◯ Other (please specify): [text field]
◯ Decline to answer

What is the highest level of education you have completed?*(mandatory question)*
◯ Some high school
◯ High school/GED
◯ Some college
◯ Associate's degree
◯ Bachelor's degree
◯ Master's degree
◯ Doctorate degree
◯ Law degree
◯ Medical degree
◯ Trade or other technical school degree
◯ Decline to answer

The power switch on a computer is used to:*(mandatory question)*
◯ Call customer support for help
◯ Install new software from a DVD

◯ Turn the computer on and off
◯ Print documents to a laser printer
◯ Run an anti-virus program
◯ Send email messages
◯ I don't know

If a macro is received from an untrusted sender in Microsoft Outlook, this means that:*(mandatory question)*
◯ The macro's memory is full
◯ An anti-virus program has detected that the macro was installed by the government
◯ The macro is not signed or it was signed with an unknown private key
◯ The macro is known to be a third party advertiser
◯ The macro is a pop up ad
◯ The macro is an unsecured piece of hardware
◯ I don't know

The main security concern caused by connecting to a server that has an untrusted certificate is:*(mandatory question)*
◯ You might be connecting to a server that has a certificate with a virus
◯ You might be connecting to a server that has a certificate that is a virus
◯ You might be allowing ads to pop up on your computer
◯ You might be sending the contents of your address book to an untrusted source
◯ You may have not plugged in your certificate properly
◯ You might not be connected to the server you want to be connected to
◯ I don't know

The best definition of cryptographically signing an email message is:*(mandatory question)*
◯ Putting up a firewall with a private key
◯ Running an anti-virus check on the message
◯ Associating one's private key with the message
◯ Sending the message over a licensed network connection
◯ Adding a footer to the message with your name
◯ Viewing the message over a secure SSL connection
◯ I don't know

The main security concern caused by not encrypting an email message is:*(mandatory question)*
◯ The email message might be read by a malicious third party
◯ The email message might be a virus
◯ A malicious third party might break the password on the email message
◯ The email message might breach your firewall
◯ The email might be violating network licensing agreements
◯ The recipient might not be able to decrypt the email message
◯ I don't know

The main security concern caused by not cryptographically signing an email message is:*(mandatory question)*

◯ If your message is modified, the recipient will be unable to detect that it has been tampered with
◯ The email message might breach your firewall
◯ A malicious third party could sign your email with your private key
◯ The email message might be sent over an unsecured connection
◯ The email message might be a virus
◯ You might not be able to track the email message
◯ I don't know

Appendix D

Experimental material for attractors study

D.1 Algorithm used for Progressive Reveal

In the *Reveal* attractor, a *target text* is faded out all at once, and then progressively faded in. Each character of the target text fades in from 0% opacity to 100% opacity in 10% increments. To raise salience, the timing of the increments is both random and non uniform, favoring English reading order (left to right). We run a new round of the darkening algorithm every 50ms, and in each round r we generate a random number $x_{r,i}$ for each character index i within the string. The character at index i becomes 10% darker if $x_{r,i} < .25 + \frac{r-2i}{L}$, were L is the length of the string. The result is an eye-catching progression in which characters are revealed mostly, but not entirely, from left to right. While we only tested this algorithm with text, a similar algorithm could be used to reveal images progressively.

D.2 Recruitment and instructions

D.2.1 Text used in Mechanical Turk HIT, Experiments 1 and 2

Researchers at Carnegie Mellon University are conducting a set of brief surveys about online games. You will have to play three online games, and then answer a short survey giving us your opinion about each game. The whole survey should take you about 20 minutes. We will pay you $1.00 for your participation.

Requisites to participate:

1. You must be 18 years old or older.

2. You must be in the United States while taking the survey.

3. You must use Microsoft Windows Vista or 7. You may use either Firefox, Chrome or Internet Explorer (in which case it has to be IE9 or higher.)

4. You must not take this survey twice. Please click here to check if you have taken this survey before, or any earlier version of this survey.

To be paid, follow these steps:

1. Go to: [URL shown here]
2. After completing the survey you will receive a confirmation code in the last page. Enter the code in the box below and we will approve your payment. Please do not enter the code more than once. If you are not sure about having entered the code correctly, please send us a message instead of trying to send the HIT twice. Please do not make up codes. If you make up a code to obtain the payment, we will reject your HIT.

Enter your code here: [Text box shown here]

For questions and problems, please contact us through Mechanical Turk's contact functionality.

D.2.2 Example of instructions to participants before each game

Instructions to evaluate the game:

1. Please click on the link below to open the game in a new window/tab of your browser.
2. Wait for the game to load. When it's fully loaded, play the game "Tom and Jerry Refrigerator Raid Game" for about 2 to 3 minutes.
3. Return to this survey to answer the questions below.

Assigned game #1: Tom and Jerry Refrigerator Raid Game [URL goes here]

Attention: By clicking on this link you acknowledge that the website you will be directed to is in no way affiliated with Carnegie Mellon University (CMU), and that CMU is in no way responsible for the content of this website.

D.3 Exit survey for Experiment 1

ALL PARTICIPANTS:

1. Since the moment you started this study, have you seen any windows that asked if you wanted to allow software to be installed on your computer?
◯ Yes
◯ No

IF ANSWERED "No" TO 1:

2. Below you will find several security windows. If you have seen one of these windows recently, or a window that is similar to one of these, please click on it.
Answer options are the actual security window, two other security windows and an option for "If you have not seen any of these warnings, please click here"

IF ANSWERED "Yes" TO 1 OR PICKED THE ACTUAL SECURITY WINDOW IN 2:

3. Do you have an antivirus software installed in your computer?
◯ Yes
◯ No
◯ I'm not sure

The questions below refer to the window that asked you whether you wanted to allow software to be installed.

4. What did you do when the window appeared on your screen? If you saw the same window multiple times, please tell us only the last thing you did.
◯ I clicked on the 'X' on the top right corner
◯ I clicked on "Cancel the installation."
◯ I clicked on "Install the software."
◯ I closed the browser's tab or the browser's window
◯ Other (please explain): [textarea]
◯ I don't remember

5. Why did you decide to install the software? [textarea]

6. What was the name of the publisher of the software to be installed? (if you are not sure, please provide your best guess)
◯ Adobe Corporation (adobe.com)
◯ Addobee Corp (adoobee.com)
◯ Microsoft Corporation (microsoft.com)
◯ Miicr0s0ft Corporation (miicr0s0ft.com)
◯ I didn't look
◯ I may have looked but have no recollection
◯ Other (Please explain): [textarea]

7. Have you seen this window before?
◯ No, I have not seen this warning before
◯ I've seen a warning that warned me about this type of problem, but the warning looked different
◯ Yes, I've seen this warning before
◯ I'm not sure

IF ANSWERED "I clicked on 'Install the software'" TO 4 AND "Addobee Corp (adoobee.com)" OR "Miicr0s0ft Corporation (miicr0s0ft.com)" TO 6 ASK THIS FOLLOW UP QUESTION:

8. In previous questions, we asked you "what did you do when the window appeared on your screen?", and your answer was "I clicked on 'Install the software.'" Later, we asked you if you recalled the publisher of the software, and your answer was [answer to 6]

Could you please explain briefly again why you decided to install the software? [textarea]

---

9. Please select the option that most accurately completes the following sentence:
"When the warning popped up, I believed it was..."
◯ "a Microsoft Internet Explorer warning"
◯ "a Microsoft Windows warning"
◯ "a fake warning"
◯ Other
◯ I'm not sure

10. Please explain your answer to the above question in as much detail as possible: [textarea]

The installation window that you saw when visiting the last website was actually part of the content of the website www.yourgamefactory.net. The website, which is run by our researchers, created a window and made it appear as if it came from your web browser. We were mimicking windows from your browser so that we could understand how you handle security decisions, such as the decision to install software. This research will be used in the design of user interfaces that help users make better security decisions.

11. At the time you saw the installation window, who did you think produced it? (In other words, who wanted to warn you. This question is not about who you were being warned about.)
◯ The gaming website
◯ A malicious attacker
◯ The browser (Chrome, Firefox or Internet Explorer)
◯ Microsoft Windows
◯ The researchers running this study
◯ I'm not sure

12. Please explain your answer to the above question in as much detail as possible: [textarea]

13. Did you think that the installation window was part of the study?
◯ Yes
◯ I'm not sure
◯ No

14. At the time you saw the installation window, did you suspect that the window was actually faked by the website?
◯ I never suspected
◯ Something felt funny or suspicious, but I had no idea what it was
◯ I suspected that the warning was faked by the website
◯ I was completely sure that the warning was faked by the website

15. Why did you suspect the warning was fake?
◯ I moved my browser window and the warning moved with it
◯ I resized my browser window and the warning was hidden
◯ I tried to move the warning window and it did not move
◯ The warning is not a warning that appears in my browser
◯ The text in the warning was different from the text I've seen before
◯ Security warnings flash three times when they first appear, whereas this warning did not flash
◯ The warning's color scheme did not match my browser's color scheme
◯ I was told that the warning would be fake
◯ Other (please explain in detail) [textarea]

IF ANSWERED "NO" TO 1 AND DID NOT PICK THE ACTUAL SECURITY WINDOW IN 2:

16. Did you visit the last gaming website `(www.yourgamefactory.net)`?
◯ Yes
◯ No

17. Please describe carefully what you did to play the game at the last website you visited (yourgamefactory.net): [textarea]

18. Please describe carefully why you did not visit the last website (yourgamefactory.net): [textarea]

ALL PARTICIPANTS:

19. Do you know any programming languages?
◯ Yes (please indicate which programming languages you know): [textarea]
◯ No

20. What is your gender?
◯ Female
◯ Male
◯ Decline to answer

21. What is your age? [dropdown]

22. What is your race/ethnicity?
◯ Asian/Pacific Islander
◯ Black/African-American
◯ White/Caucasian
◯ Hispanic
◯ Native American/Alaska Native
◯ Other/Multi-Racial
◯ Decline to answer

23. What is your current occupation?
◯ Administrative Support (eg., secretary, assistant)
◯ Art, Writing and Journalism (eg., author, reporter, sculptor)
◯ Business, Management and Financial (eg., manager, accountant, banker)
◯ Education (eg., teacher, professor)
◯ Legal (eg., lawyer, law clerk)
◯ Medical (eg., doctor, nurse, dentist)
◯ Science, Engineering and IT professional (eg., researcher, programmer, IT consultant)
◯ Service (eg., retail clerks, server)
◯ Skilled Labor (eg., electrician, plumber, carpenter)
◯ Student
◯ Other Professional
◯ Not Currently Working/Currently Unemployed
◯ Retired
◯ Other (please specify): [textarea]
◯ Decline to answer

24. What is the highest level of education you have completed?
◯ Some high school
◯ High school/GED
◯ Some college
◯ Associate's degree
◯ Bachelor's degree
◯ Master's degree
◯ Doctorate degree
◯ Law degree
◯ Medical degree
◯ Trade or other technical school degree
◯ Decline to answer

D.4 Exit survey for Experiment 2

[Only questions that are different from the exit survey in Experiment 1 are included below.]

1. Since the moment you started this study, have you seen a pop-up window that asked for upgraded permissions on your computer?
◯ Yes
◯ No

4. What did you do when the pop-up window appeared on your screen? If you saw the same window multiple times, please tell us only the last thing you did.
◯ I clicked on the 'X' on the top right corner
◯ I clicked on "No, do not run the application"
◯ I clicked on "Yes, run the application with upgraded permissions"
◯ I closed the browser's tab or the browser's window
◯ Other (please explain): [textarea]
◯ I don't remember

5. Why did you decide to run the application with upgraded permissions? [textarea]

6. What upgraded permissions was the website requesting? (if you are not sure, please provide your best guess)
◯ A website cookie
◯ All files and folder on the computer
◯ The computer's screen
◯ The browser's timer
◯ I didn't look
◯ I may have looked but have no recollection
◯ Other (Please explain): [textarea]

8. In previous questions, we asked you "what did you do when the window appeared on your screen?", and your answer was "I clicked on 'Yes, run the application with upgraded permissions'." Later, we asked you if you recalled the resource that was being requested, and your answer was [answer to 6].

Could you please explain briefly again why you decided to run the application? [textarea]

9. Please select the option that most accurately completes the following sentence:
"When the pop-up window appeared on my screen, I believed it was..."
◯ "displayed by Adobe"
◯ "displayed by my browser"
◯ "displayed by Microsoft Windows"
◯ "displayed by a virus or malware"
◯ Other
◯ I'm not sure

13. Did you think that the pop-up window was part of the study?
◯ Yes

◯ I'm not sure
◯ No

14. At the time you saw the pop-up window, did you suspect that the window was actually faked by the website?
◯ I never suspected
◯ Something felt funny or suspicious, but I had no idea what it was
◯ I suspected that the warning was faked by the website
◯ I was completely sure that the warning was faked by the website

D.5 Debrief questions

In studies 1 and 2 we presented the debrief text below to all of our participants at the very end of the exit survey, as mandated by our Institutional Review Board. In addition, we asked the question in Section D.5.2 below to approximately two thirds of our participants, and the questions in Section D.5.3 to the last third of our participants.

D.5.1 Debrief text presented to all participants

About this survey (please read!)

Thank you for participating. Below you will find some important details about this research.

Online games websites are notorious for having viruses. Please be assured that we sent you only to reputable websites. If you saw a warning on one of these websites, that was a test warning that we inserted as part of this study. You were not actually in any danger.

Computer security dialogs are an important part of almost every computer program today. Their purpose is to protect your computer and the information stored in your computer from risks like viruses, malware, and online fraud. However important, computer security dialogs can sometimes be difficult to understand. Through this research, we hope to develop guidelines to help improve computer security dialogs so that they will be more useful and better protect users.

If you want to know more about computer warnings and their importance, please consult the links and articles that we have included below. If you have any concerns, please do not hesitate to contact us: Cristian Bravo-Lillo, `cbravo@cmu.edu`, CyLab Usable Privacy and Security Laboratory, Carnegie Mellon University, Pittsburgh, PA, USA.

Thanks again for participating in our research.

[References to papers and educational material online included here.]

D.5.2 First version of debrief questions

In order to capture people's natural behavior, it is sometimes necessary for researchers to deceive study participants. This study contained a number of elements of deception.

First, the study was not actually about online games. Second, the website of the third game in the study (`yourgamefactory.net`) was not a 'third-party' site, but is actually operated by our researchers. Third, the website did not actually need to install or update Silverlight on your computer. Finally, the installation window that popped over that website, which appeared to be from

Microsoft Windows, was actually an imitation created by the webpage. No software was actually being downloaded and no software would be installed, even if you chose the option to install.

As researchers, we take the safety of our participants very seriously and we are required to minimize the risk you undertake by participating in the study. No software was actually installed in this study, even when participants believed they were allowing software to be installed on their computers so that they could run a game. As part of our obligation to protect the safety of our participants, we submitted our study for review by Carnegie Mellon University's institutional review board (also known as an ethics board), which approved our research.

However, if you feel the study has caused you harm; if you feel the use of deception was unwarranted, unethical, or otherwise unacceptable; or if you have any other concerns with how this study was run, please share your concerns with us below:
[Free response included here.]

D.5.3 Second version of debrief questions

In this experiment we measured how different techniques for presenting information help users to make security decisions. We hope that the results of this study will lead to improvements in the security of computing systems and benefit those who use them.

One challenge in studying security decision making is that if participants are made aware (or become aware) that researchers are studying their security behavior, they are more likely to pay attention to security than they would normally. In order to capture people's natural behavior, it is sometimes necessary for researchers to deceive study participants. This study contained a number of elements of deception.

First, the study was not actually about online games. Second, the website of the third game in the study (`yourgamefactory.net`) was not a 'third-party' site, but is actually operated by our researchers. Third, the website did not actually need to install or update Silverlight on your computer. Finally, the installation window that popped over that website, which appeared to be from Microsoft Windows, was actually an imitation created by the web page. No software was downloaded and no software was installed, even if you chose the option to install.

As researchers, we take the safety of our participants very seriously, and we are required to minimize the risk you undertake by participating in the study. No software was actually installed in this study, even when participants believed they were allowing software to be installed on their computers so that they could run a game. As part of our obligation to protect the safety of our participants, we submitted our study for review by Carnegie Mellon University's institutional review board (also known as an ethics board), which approved our research.

We would like to solicit your feedback for help in evaluating the ethical acceptability of this research study, and to use your feedback to inform decisions to permit or disallow similar studies in the future.

Do you believe this experiment should be allowed to proceed, or do you feel that the potential risk of harm outweighs the potential benefit to computer security researchers and society as a whole?
◯ This experiment should definitely be allowed to proceed.
◯ This experiment should probably be allowed to proceed, but with caution.
◯ This experiment should probably not be allowed to proceed.
◯ This experiment should definitely not be allowed to proceed.

Please explain why you believe the experiment should or should not be allowed to proceed: **[Free response included here.]**

Appendix E

Experimental material for habituation study

E.1 Text used in Mechanical Turk HIT in Experiments 1 and 2

Researchers at Carnegie Mellon University are conducting a set of experiments with pop-up dialogs. You will have to repeat a task for 5 minutes, and then answer a short survey. The whole study should take you about 10 minutes. We will pay you $0.50 for your participation.

Requisites to participate:
[Same requisites than in attractors study go here.]

E.2 Instructions given to participants in Experiments 1 and 2

In the following page you will see a timer on the screen, and a number of consecutive dialogs (pop-up windows) asking you to click 'Yes' or 'No'. **Your task is to respond to as many dialogs as you can before the timer goes off**. You can increase your performance by following instructions and responding to each question quickly. Some dialogs may require you to wait or perform an action before the 'Yes' button is activated.

Those who perform well may be rewarded with opportunities to finish the study early while still receiving their full payment. After finishing the task, you will have to answer a short survey.

When you are ready to begin, please click on the URL below. [URL shown.]

E.3 Exit survey for Experiment 1

1. The image below corresponds to one of the dialogs you saw during this study: [image shown]

Please type in the contents of the “Status:” field in the most-recently shown dialog, to the best of your memory. If you have no memory, please type “none”: [textarea shown]

---

2. What did the last status message you saw communicate?
◯ That I should press “yes” to continue with the study
◯ That I could press “no” to finish the study early
◯ The number of messages that I dismissed
◯ The amount of money I will be paid for this study
◯ That I could press the back button to finish the study early
◯ The quality of my performance in this study
◯ I’m not sure

3. How many times did you see this message?
◯ Just once
◯ Between 2 and 4
◯ Between 5 and 8
◯ 9 or more
◯ I don’t have any recollection

[If answered ‘That I could press “no” to finish the study early’ to 2, and answered any other but ‘Just once’ to 3]

4. Why did you not press ”No” to finish the study early? [textarea shown]

---

5. Overall, how annoying was this task?
[Answers were likert-type with 5 points, from ‘Not annoying at all’ to ‘Very annoying’]

6. Did you suspect that the study may require you to answer questions about the content of the “Status” field?
◯ Definitely
◯ Somewhat
◯ Maybe a little
◯ Definitely not

7. During most of the dialogs you saw, did you intentionally read the text in the field labeled “Status”?
◯ I ignored it
◯ I tried to read a little
◯ I read every word

8. During the last dialog you saw, did you intentionally read the text in the field labeled “Status”?
◯ I ignored it
◯ I tried to read a little
◯ I read every word

9. Did you recognize that the text in the most-recently shown dialog was an instruction from the study, or did you assume it was as meaningless as the other phrases that appeared in this field?
◯ I didn’t read enough to wonder
◯ I assumed it was meaningless
◯ I recognized it was a study instruction
◯ I wasn’t sure

10. Please let us know what, if anything, was not working with the dialogs that popped up on your browser: [textarea shown]

[Questions 11 to 16 are the same as questions 19 to 24 in the Exit survey of Experiments 1 and 2.]

Appendix F

Experimental material for factors study

F.1 Text used in Mechanical Turk HIT

Researchers at Carnegie Mellon University are conducting a set of brief surveys about online games. You will have to play three online games, and then answer a short survey giving us your opinion about each game. The whole survey should take you about 20 minutes. We will pay you $1.00 for your participation.
Requisites to participate:

1. You must be 18 years old or older.
2. You must use Microsoft Windows Vista or superior. You may use either Firefox, Chrome or Internet Explorer (in which case it has to be IE9 or higher.)
3. You must not take this survey twice.Please click here to check if you have taken this survey before, or any earlier version of this survey.

To be paid, follow these steps:

1. Go to: [URL shown here]
2. Complete the survey until you receive confirmation that you completed the survey.
3. After you complete the survey, please confirm that you completed the survey below and click on the button 'Submit':
   [Checkbox] I completed this survey as thoroughly as possible to me. I accept that my HIT be rejected if I did not do the work or if I tried to cheat.

For questions and problems, please contact us through Mechanical Turk's contact functionality.

F.2 Exit survey

The questions below refer to the window that asked you whether you wanted to allow software to be installed.

1. What did you do when the window appeared on your screen? If you saw the same window multiple times, please tell us only the last thing you did.

   ◯ I clicked on the 'X' on the top right corner.
   ◯ I clicked on "Cancel the installation."
   ◯ I clicked on "Install the software."
   ◯ I closed the browser's tab or the browser's window.
   ◯ I don't remember
   ◯ Other (please explain)

2. What was the name of the publisher of the software to be installed? (if you are not sure, please provide your best guess)

   ◯ Microsoft Corporation (microsoft.com)
   ◯ Addobee Corp (adoobee.com)
   ◯ Adobe Corporation (adobe.com)
   ◯ Miicr0s0ft Corporation (miicr0s0ft.com)
   ◯ Other
   ◯ I didn't look
   ◯ I may have looked but have no recollection

3. In previous questions, we asked you *"What did you do when the window appeared on your screen?"*, and your answer was *"I clicked on 'Install the software'."* Later, we asked you if you recalled the publisher of the software, and your answer was _____.

   Could you please explain why you decided to install the software even after recognizing the odd-looking publisher name? [Textare shown here.]

4. Please select the option that most accurately completes the following sentence: *"When the warning popped up, I believed it was _____"*

   ◯ "a warning from my Browser (Internet Explorer, Chrome, or Firefox)"
   ◯ "a warning from Microsoft Windows"
   ◯ "a fake warning"
   ◯ Other
   ◯ I'm not sure

5. Please explain your answer to the above question. [Textarea shown here.]

6. The installation window that you saw when visiting the last website was actually part of the content of the website www.yourgamefactory.net. The website, which is run by our researchers, created a window and made it appear as if it came from your web browser. We were mimicking windows from your browser so that we could understand how you handle

security decisions, such as the decision to install software. This research will be used in the design of user interfaces that help users make better security decisions.

At the time you saw the installation window, who did you think produced it? *(In other words, who wanted to notify you that software was being installed. This question is not about the publisher of the software.)*

◯ A malicious attacker
◯ The browser (Chrome, Firefox or Internet Explorer)
◯ The researchers running this study
◯ The gaming website
◯ Microsoft Windows
◯ I'm not sure

7. Please explain your answer to the above question in as much detail as possible: [Textarea shown here.]

8. Please indicate how much do you agree with the following sentences: [Each of the items below has a 5-points Likert-type scale, from 'I completely disagree' to 'I completely agree'.]

   - "It would be a major hassle for me if the computer I am using for this study got infected with a virus."
   - "If the computer I am using for this study got infected with a virus, that would take a lot of effort to fix."
   - "It would be a waste of time if the computer I am using for this study got infected with a virus."
   - "I would feel very embarrassed if the computer I am using for this study got infected with a virus."

9. Do you have antivirus software installed on the computer you used to perform this experiment?

   ◯ No
   ◯ I'm not sure
   ◯ Yes (Please indicate which software you have)

10. Please indicate whether you believe an antivirus software can protect you and/or your computer from the following threats: [Each of the items below has a 4-points Likert-type scale: 'Never', 'Occasionally', 'Mostly', and 'Always'.]

    - Spam email
    - Malware
    - Phishing

- Identity theft
- Online stalking
- Fake antivirus software
- Purchase frauds

11. Why have you not installed antivirus software on your computer? [This question is shown only when the answer to 'Do you have antivirus...' is 'No']

12. Are you at all concerned about not having antivirus on your computer? [This question is shown only when the answer to 'Do you have antivirus...' is 'No']

13. Is the antivirus software in this computer configured to automatically update itself to protect against the latest threats? [This question is shown only when the answer to 'Do you have antivirus...' is 'Yes']

    ◯ No
    ◯ I'm not sure
    ◯ Yes

14. When was the last time that antivirus software was updated in this computer? [This question is shown only when the answer to 'Do you have antivirus...' is 'Yes']

    ◯ Today or Yesterday
    ◯ Less than a week ago
    ◯ Less than two weeks ago
    ◯ Less than a month ago
    ◯ More than a month ago
    ◯ I have never updated my antivirus software
    ◯ I don't know

15. Please indicate how confident you are that your antivirus software is protecting you and/or your computer from the following threats: [This question is shown only when the answer to 'Do you have antivirus...' is 'Yes'. The options to this question are the same as the options to the question 'Please indicate whether you believe an antivirus software can protect you and/or your computer...' above. The answers to each option are a 5-points Likert-type scale: 'Not applicable', 'Not confident at all', 'Slightly confident', 'Very confident', and 'Completely confident']

[Same questions as in exit survey for Attractors study go here. See questions 19 through 24 in Appendix E.3]

Bibliography

[1] Andre Adelsbach, Sebastian Gajek, and Jorg Schwenk. Visual spoofing of SSL protected web sites and effective countermeasures. *Information Security Practice and Experience*, pages 204–216, 2005.

[2] Devdatta Akhawe and Adrienne Porter Felt. Alice in warningland: A large-scale field study of browser security warning effectiveness. In *Proceedings of the 22th USENIX Security Symposium*, 2013.

[3] John R. Anderson. *Cognitive Psychology and Its Implications*. Worth Publishers, New York, 6th edition edition, 2005.

[4] Ross Anderson, Chris Barton, Rainer Böhme, Richard Clayton, Michael van Eeten, Michael Levi, Tyler Moore, and Stefan Savage. Measuring the cost of cybercrime. In *11th annual Workshop on the Economics of Information Security*, 25 June 2012.

[5] Apple Inc. Apple human interface guidelines. Online document available at http://developer.apple.com, 2010. Last visit on Apr/08/2010.

[6] G. Susanne Bahr and Richard A. Ford. How and why pop-ups don't work: Pop-up prompted eye movements, user affect and decision making. *Computers in Human Behavior*, 27(2):776–783, 2011.

[7] Steffen Bartsch, Melanie Volkamer, Heike Theuerling, and Fatih Karayumak. Contextualized web warnings, and how they cause distrust. In Michael Huth, N. Asokan, Srdjan apkun, Ivan Flechais, and Lizzie Coles-Kemp, editors, *Trust and Trustworthy Computing*, volume 7904 of *Lecture Notes in Computer Science*, pages 205–222. Springer Berlin Heidelberg, 2013.

[8] T. Randolph Beard, George S. Ford, Thomas M. Koutsky, and Lawrence J. Spiwak. Tort liability for software developers: A law & economics perspective. *J. Marshall J. Computer & Info. L.*, 27:199–613, 2009.

[9] Calum Benson, Adam Elman, Seth Nickell, and Colin Z. Robertson. Gnome human interface guidelines 2.2.1. Online document available at http://library.gnome.org, 2010. Last visit on Apr/08/2010.

[10] Robert Biddle, Paul C van Oorschot, Andrew S Patrick, Jennifer Sobey, and Tara Whalen. Browser interfaces and extended validation ssl certificates: an empirical study. In *Proceedings of the 2009 ACM workshop on Cloud computing security*, pages 19–30. ACM, 2009.

[11] Rainer Böhme and Stefan Köpsell. Trained to accept?: a field experiment on consent dialogs. In Elizabeth D. Mynatt, Don Schoner, Geraldine Fitzpatrick, Scott E. Hudson, W. Keith Edwards, and Tom Rodden, editors, *CHI*, pages 2403–2406. ACM, 2010.

[12] Susana Rankin Bohme and David Egilman. Consider the Source: Warnings and Anti-Warnings in the Tobacco, Automobile, Beryllium, and Pharmaceutical Industries. In Michael S. Wogalter, editor, *Handbook of warnings*, chapter 51, pages 635–644. Lawrence Erlbaum Associates, Mahwah, New Jersey, 2006.

[13] Cristian Bravo-Lillo, Lorrie F. Cranor, Julie Downs, and Saranga Komanduri. Bridging the Gap in Computer Security Warnings: A Mental Model Approach. *IEEE Security & Privacy Magazine*, 9(2):18–26, March 2011.

[14] Cristian Bravo-Lillo, Lorrie F. Cranor, Julie Downs, Saranga Komanduri, Rob Reeder, Stuart Schechter, and Manya Sleeper. Your Attention Please: Designing security-decision UIs for users habituated to ignoring them. In *Proceedings of the Ninth Symposium on Usable Privacy and Security*, SOUPS '13, pages 6:1–6:12, New York, NY, USA, 2013. ACM.

[15] Cristian Bravo-Lillo, Lorrie F. Cranor, Julie Downs, Saranga Komanduri, Stuart Schechter, and Manya Sleeper. Operating System Framed in Case of Mistaken Identity. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, ACM CCS'12, October 2012.

[16] Cristian Bravo-Lillo, Lorrie F. Cranor, Julie Downs, Saranga Komanduri, and Manya Sleeper. Improving Computer Security Dialogs. In Pedro Campos, Nicholas Graham, Joaquim Jorge, Nuno Nunes, Philippe Palanque, and Marco Winckler, editors, *Lecture Notes in Computer Science*, volume 6949 of *Lecture Notes in Computer Science*, chapter 2, pages 18–35. Springer, Berlin, Heidelberg, 2011.

[17] Shlomo Breznitz. *Cry wolf: The psychology of false alarms*. Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1984.

[18] José C. Brustoloni and Ricardo V. Salomón. Improving security decisions with polymorphic and audited dialogs. In *Proceedings of the Third Symposium on Usable Privacy and Security*, SOUPS '07, pages 76–85, New York, NY, USA, 2007. ACM.

[19] L. Jean Camp. Mental Models of Computer Security. In Ari Juels, editor, *Financial Cryptography*, volume 3110 of *Lecture Notes in Computer Science*, pages 106–111. Springer, 2004.

[20] Microsoft Corporation. What is user account control? `http://windows.microsoft.com/en-US/windows-vista/What-is-User-Account-Control`.

[21] Marco Cova. Personal correspondence with Stuart Schechter, May 2012.

[22] Marco Cova, Corrado Leita, Olivier Thonnard, Angelos D. Keromytis, and Marc Dacier. An analysis of rogue AV campaigns. In *Proceedings of the 13th International Symposium on Recent Advances in Intrusion Detection (RAID 2010)*, pages 442–463, September 2010.

[23] Eli P. Cox III. Marketing versus Warning. In Michael S. Wogalter, editor, *Handbook of warnings*, chapter 52, pages 645–652. Lawrence Erlbaum Associates, Mahwah, New Jersey, 2006.

[24] Lorrie F. Cranor. A framework for reasoning about the human in the loop. In *A framework for reasoning about the human in the loop*, UPSEC'08, pages 1–15, Berkeley, CA, USA, 2008. USENIX Association.

[25] Michael A Cusumano. Who is liable for bugs and security flaws in software? *Communications of the ACM*, 47(3):25–27, 2004.

[26] Rachna Dhamija and J. D. Tygar. The battle against phishing: Dynamic security skins. In *Proceedings of the 2005 Symposium on Usable Privacy and Security*, SOUPS '05, pages 77–88, New York, NY, USA, 2005. ACM.

[27] Rachna Dhamija, J. D. Tygar, and Marti Hearst. Why phishing works. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, pages 581–590, New York, NY, USA, 2006. ACM.

[28] Rachna Dhamija and J.D. Tygar. Phish and hips: Human interactive proofs to detect phishing attacks. In HenryS. Baird and DanielP. Lopresti, editors, *Human Interactive Proofs*, volume 3517 of *Lecture Notes in Computer Science*, pages 127–141. Springer Berlin Heidelberg, 2005.

[29] Julie S. Downs, Mandy B. Holbrook, and Lorrie Faith Cranor. Decision strategies and susceptibility to phishing. In *Proceedings of the Second Symposium on Usable Privacy and Security*, volume 149 of *SOUPS '06*, pages 79–90, New York, NY, USA, 2006. ACM.

[30] Julie S. Downs, Mandy B. Holbrook, Steve Sheng, and Lorrie F. Cranor. Are your participants gaming the system?: screening mechanical turk workers. In *Proceedings of the ACM Computer-Human Interaction Conference 2010*, CHI '10, pages 2399–2402, New York, NY, USA, 2010. ACM.

[31] W. Keith Edwards, Erika S. Poole, and Jennifer Stoll. Security automation considered harmful? In *Proceedings of the 2007 Workshop on New Security Paradigms*, NSPW '07, pages 33–42, New York, NY, USA, 2007. ACM.

[32] Judy Edworthy and Austin Adams. *Warning Design: A Research Prospective*. Taylor & Francis, 1996.

[33] Serge Egelman. *Trust Me: Design Patterns for Constructing Trustworthy Trust Indicators*. PhD thesis, School of Computer Science, April 2009.

[34] Serge Egelman, Lorrie Cranor, and Jason Hong. You've been warned: an empirical study of the effectiveness of web browser phishing warnings. In *Proceedings of the ACM Computer-Human Interaction Conference 2008*, CHI '08, pages 1065–1074, New York, NY, USA, 2008. ACM.

[35] David Egilman and Susana Rankin Bohme. A brief history of warnings. In Michael S. Wogalter, editor, *Handbook of warnings*, chapter 2, pages 11–20. Lawrence Erlbaum Associates, Mahwah, New Jersey, 2006.

[36] Edward W. Felten, Dirk Balfanz, Drew Dean, and Dan S. Wallach. Web spoofing: An Internet con game. In *20th National Information Systems Security Conference*, October 1996.

[37] Norman Feske and Christian Helmuth. A nitpicker's guide to a minimal-complexity secure GUI. In *Proceedings of the 21st Annual Computer Security Applications Conference*, pages 85–94, Washington, DC, USA, 2005. IEEE Computer Society.

[38] Dinei Florencio and Cormac Herley. Where do all the attacks go? In Bruce Schneier, editor, *Economics of Information Security and Privacy III*, pages 13–33. Springer New York, 2013.

[39] Google's income statement information. `http://investor.google.com/financial/tables.html`, 2013. Retrieved on Feb/01/2014.

[40] Google's safe browsing. `https://developers.google.com/safe-browsing/`, 2013. Retrieved on Dec/30/2013.

[41] Cormac Herley. So long, and no thanks for the externalities: the rational rejection of security advice by users. In *Proceedings of the New Security Paradigms Workshop 2009*, NSPW '09, pages 133–144, New York, NY, USA, 2009. ACM.

[42] Amir Herzberg and Ahmad Gbara. Security and identification indicators for browsers against spoofing and phishing attacks. Cryptology ePrint Archive, Report 2004/155, 2004. `http://eprint.iacr.org/`.

[43] Internal revenue service, instructions to form 1120 for 2012. `http://www.irs.gov/pub/irs-pdf/i1120.pdf`, 2012. Retrieved on Feb/12/2014.

[44] International Telecommunication Union statistics page. `http://www.itu.int/en/ITU-D/Statistics/Pages/stat/default.aspx`, 2013. Retrieved on Dec/08/2013.

[45] Collin Jackson, Daniel R. Simon, Desney S. Tan, and Adam Barth. An evaluation of extended validation and picture-in-picture phishing attacks. In *Proceedings of the 11th International Conference on Financial Cryptography and 1st International Conference on Usable Security*, FC'07/USEC'07, pages 281–293, Berlin, Heidelberg, 2007. Springer-Verlag.

[46] Michael J. Kalsher and Kevin J. Williams. Behavioral compliance: theory, methodology, and results. In Michael S. Wogalter, editor, *Handbook of warnings*, chapter 23, pages 313–329. Lawrence Erlbaum Associates, Mahwah, New Jersey, 2006.

[47] Chris Karlof, J. D. Tygar, and David Wagner. Conditioned-safe ceremonies and a user study of an application to web authentication. In Lorrie Faith Cranor, editor, *SOUPS*, ACM International Conference Proceeding Series. ACM, 2009.

[48] Kenny Kerr. Defend your apps and critical user info with defensive coding techniques. *MSDN Magazine*, November 2004. http://msdn.microsoft.com/en-us/magazine/cc163883.aspx.

[49] Frederik Keukelaere, Sachiko Yoshihama, Scott Trent, Yu Zhang, Lin Luo, and MaryEllen Zurko. Adaptive security dialogs for improved security behavior of users. In Tom Gross, Jan Gulliksen, Paula Kotz, Lars Oestreicher, Philippe Palanque, RaquelOliveira Prates, and Marco Winckler, editors, *Human-Computer Interaction INTERACT 2009*, volume 5726 of *Lecture Notes in Computer Science*, pages 510–523. Springer Berlin Heidelberg, 2009.

[50] Soyun Kim and Michael S. Wogalter. Habituation, Dishabituation, and Recovery Effects in Visual Warnings. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 53(20):1612–1616, 2009.

[51] K. Krol, M. Moroz, and M. A. Sasse. Don't work. can't work? Why it's time to rethink security warnings. In *7th international conference on Risk and security of internet and systems (crisis)*, pages 1–8, 2012.

[52] Ponnurangam Kumaraguru, Justin Cranshaw, Alessandro Acquisti, Lorrie Cranor, Jason Hong, Mary Ann Blair, and Theodore Pham. A real-word evaluation of anti-phishing training. Technical report, Carnegie Mellon University, 2009.

[53] Kenneth R. Laughery and Michael S. Wogalter. Designing Effective Warnings. *Reviews of Human Factors and Ergonomics*, 2(1):241–271, 2006.

[54] Serge Lefranc and David Naccache. Cut-&-paste attacks with java. In *Proceedings of the 5th International Conference on Information Security and Cryptology*, ICISC'02, pages 1–15, Berlin, Heidelberg, 2003. Springer-Verlag.

[55] Mark R. Lehto. Optimal warnings: an information and decision theoretic perspective. In Michael S. Wogalter, editor, *Handbook of warnings*, chapter 7, pages 89–108. Lawrence Erlbaum Associates, Mahwah, New Jersey, 2006.

[56] Tie-Yan Li and Yongdong Wu. Trust on web browser: Attack vs. defense. In Jianying Zhou, Moti Yung, and Yongfei Han, editors, *Applied Cryptography and Network Security*, volume 2846 of *Lecture Notes in Computer Science*, pages 241–253. Springer Berlin / Heidelberg, 2003. 10.1007/978-3-540-45203-4_19.

[57] Alana Libonati, Jonathan M. McCune, and Michael K. Reiter. Usability testing a malware-resistant input mechanism. In *Proceedings of the 18th Annual Network & Distributed System Security Symposium (NDSS11)*, February 2011.

[58] M. Stuart Madden. The Duty to Warn in Products Liability. In Michael S. Wogalter, editor, *Handbook of warnings*, chapter 45, pages 583–588. Lawrence Erlbaum Associates, Mahwah, New Jersey, 2006.

[59] M. Stuart Madden. The Quiet Revolution in Post-Sale Duties. In Michael S. Wogalter, editor, *Handbook of warnings*, chapter 46, pages 589–595. Lawrence Erlbaum Associates, Mahwah, New Jersey, 2006.

[60] Max-Emanuel Maurer, Alexander De Luca, and Sylvia Kempe. Using data type based security alert dialogs to raise online security awareness. In *Proceedings of the Seventh Symposium on Usable Privacy and Security*, SOUPS '11, pages 2:1–2:13, New York, NY, USA, 2011. ACM.

[61] Joachim Meyer. Responses to Dynamic Warnings. In Michael S. Wogalter, editor, *Handbook of warnings*, chapter 16, pages 221–230. Lawrence Erlbaum Associates, Mahwah, New Jersey, 2006.

[62] Microsoft Corporation. Windows user experience interaction guidelines. Online document available at http://msdn.microsoft.com, 2010. Retrieved on Apr/08/2010.

[63] Granger M. Morgan, Baruch Fischhoff, Ann Bostrom, and Cynthia J. Atman. *Risk Communication: A Mental Models Approach*. Cambridge University Press, 2001.

[64] Sara Motiee, Kirstie Hawkey, and Konstantin Beznosov. Do windows users follow the principle of least privilege?: investigating user account control practices. In *Proceedings of the Sixth Symposium on Usable Privacy and Security*, SOUPS '10, pages 1:1–1:13, New York, NY, USA, 2010. ACM.

[65] Initializing Winlogin. `http://msdn.microsoft.com/en-us/library/windows/desktop/aa375994`, 2012. Retrieved on Dec/11/2013.

[66] Chris Nodder. Users and trust: A microsoft case study. In Lorrie Faith Cranor and Simson L. Garfinkel, editors, *Security and Usability: Designing Secure Systems That People Can Use*, Theory in practice, chapter 29, pages 589–606. O'Reilly Media, Inc., Sebastopol, CA, USA, first edition, 2005.

[67] Bryan Parno, Cynthia Kuo, and Adrian Perrig. Phoolproof phishing prevention. In *Proceedings of the Financial Cryptography and Data Security 10th International Conference*, FC'06, 2006.

[68] Geoffrey M. Peckham. An Overview of the ANSI Z535 Standards for Safety Signs, Labels, and Tags. In Michael S. Wogalter, editor, *Handbook of warnings*, chapter 33, pages 437–444. Lawrence Erlbaum Associates, Mahwah, New Jersey, 2006.

[69] Lee Rainie, Sara Kiesler, Ruogu Kang, and Mary Madden. Anonymity, privacy, and security online. http://pewinternet.org/Reports/2013/Anonymity-online.aspx, 2013.

[70] Moheeb Abu Rajab, Lucas Ballard, Panayiotis Mavrommatis, Niels Provos, and Xin Zhao. The nocebo* effect on the web: An analysis of fake anti-virus distribution. In *Proceedings of the 3rd USENIX Conference on Large-Scale Exploits and Emergent Threats: Botnets, Spyware, Worms, and More*, LEET'10, pages 3–3, Berkeley, CA, USA, 2010. USENIX Association.

[71] Robert W. Reeder, Ellen Cram Kowalczyk, and Adam Shostack. NEAT effective warnings. `http://www.microsoft.com/en-us/download/details.aspx?id=34958`, 2011. Accessed: 03/26/2013.

[72] Consumer Product Warnings: Research and Recommendations. Responses to Dynamic Warnings. In Michael S. Wogalter, editor, *Handbook of warnings*, chapter 10, pages 137–146. Lawrence Erlbaum Associates, Mahwah, New Jersey, 2006.

[73] Robert Rosenthal and Ralph L Rosnow. *Essentials of Behavioral Research: Methods and Data Analysis*. McGraw-Hill, New York, USA, 3rd edition edition, 2008.

[74] Blake Ross. Firefox and the Worry-Free Web. In Lorrie Faith Cranor and Simson L. Garfinkel, editors, *Security and Usability: Designing Secure Systems That People Can Use*, Theory in practice, chapter 28, pages 589–606. O'Reilly Media, Inc., Sebastopol, CA, USA, first edition, August 2005.

[75] Blake Ross, Collin Jackson, Nicholas Miyake, Dan Boneh, and John C. Mitchell. Stronger password authentication using browser extensions. In *Proceedings of the Proceedings of the 14th Usenix Security Symposium*, August 2005.

[76] Jerome H Saltzer and Michael D Schroeder. The protection of information in computer systems. *Proceedings of the IEEE*, 63(9):1278–1308, 1975.

[77] Roger C Schank and Robert P Abelson. *Scripts, plans, goals, and understanding: An inquiry into human knowledge structures*. Psychology Press, 2013.

[78] Stuart Schechter, Rachna Dhamija, Andy Ozment, and Ian Fischer. The Emperor's New Security Indicators. In *Proceedings of the IEEE Symposium on Security and Privacy*, IEEE SP '07, pages 51–65, Washington, DC, USA, 2007. IEEE.

[79] "Security-on-a-Stick" to protect consumers and banks from the most sophisticated hacker attacks. `http://www.zurich.ibm.com/news/08/ztic.html`, October 2008.

[80] Jonathan S. Shapiro, John Vanderburgh, Eric Northup, and David Chizmadia. Design of the EROS trusted window system. In *Proceedings of the 13th Conference on USENIX Security Symposium*, SSYM'04, pages 12–12, Berkeley, CA, USA, 2004. USENIX Association.

[81] David Sharek, Cameron Swofford, and Michael Wogalter. Failure to Recognize Fake Internet Popup Warning Messages. *Proc. of Human Factors and Ergonomics Society*, 52(6):557–560, 2008.

[82] Steve Sheng, Lorrie Faith Cranor, Jason I. Hong, Brad Wardman, Gary Warner, and Chengshan Zhang. An empirical analysis of phishing blacklists. In *Sixth Conference on Email and Anti-Spam*, July 2009.

[83] Jennifer Sobey, Robert Biddle, P. C. van Oorschot, and Andrew S. Patrick. Exploring User Reactions to Browser Cues for Extended Validation Certificates. Technical report, Carleton University, May 2008.

[84] Jennifer Sobey, P. C. Van Oorschot, and Andrew A. Patrick. Browser Interfaces and EV-SSL Certificates: Confusion, Inconsistencies and HCI Challenges. Technical report, Carleton University, January 2009.

[85] Andreas Sotirakopoulos, Kirstie Hawkey, and Konstantin Beznosov. On the challenges in usable security lab studies: lessons learned from replicating a study on SSL warnings. In *Proceedings of the ACM Symposium on Usable Privacy and Security 2011*, SOUPS '11, pages 3:1–3:18, New York, NY, USA, 2011. ACM.

[86] Brett Stone-Gross, Ryan Abman, Richard A. Kemmerer, Christopher Kruegel, Douglas G. Steigerwald, and Giovanni Vigna. The underground economy of fake antivirus software. In *Workshop on Economics of Information Security (WEIS)*, June 2011.

[87] Joshua Sunshine, Serge Egelman, Hazim Almuhimedi, Neha Atri, and Lorrie Cranor. Crying Wolf: An Empirical Study of SSL Warning Effectiveness. In *Proceedings of USENIX 2009 conference*, USENIX '09, 2009.

[88] Symantec Corporation. Symantec report on rogue security software, October 2009.

[89] Mohsen Tavakol and Reg Dennick. Making sense of cronbach's alpha. *International Journal of Medical Education*, 2:53–55, 2011.

[90] Rahul Telang and Sunil Wattal. An empirical analysis of the impact of software vulnerability announcements on firm stock price. *IEEE Transactions on Software Engineering*, 33(8):544–557, 2007.

[91] J. D. Tygar and Alma Whitten. WWW electronic commerce and Java trojan horses. In *Proceedings of the Second USENIX Workshop on Electronic Commerce*, volume 2, pages 15–15, Berkeley, CA, USA, 1996. USENIX Association.

[92] Alison G. Vredenburgh and Ilene B. Zackowitz. Expectations. In Michael S. Wogalter, editor, *Handbook of warnings*, chapter 25, pages 345–353. Lawrence Erlbaum Associates, Mahwah, New Jersey, 2006.

[93] Rick Wash. Folk models of home computer security. In *SOUPS '10: Proceedings of the Sixth Symposium on Usable Privacy and Security*, pages 1–16, New York, NY, USA, 2010. ACM.

[94] Tara Whalen and Kori M. Inkpen. Gathering evidence: use of visual security cues in web browsers. In Kori Inkpen and Michiel van de Panne, editors, *Graphics Interface*, pages 137–144. Canadian Human-Computer Communications Society, 2005.

[95] Alma Whitten and J Doug Tygar. Why johnny cant encrypt: A usability evaluation of pgp 5.0. In *Proceedings of the 8th USENIX Security Symposium*, volume 99. McGraw-Hill, 1999.

[96] Kim Witte. Putting the fear back into fear appeals: The extended parallel process model. *Communication Monographs*, 59(4):329–349, 1992.

[97] Michael S. Wogalter. Communication-Human Information Processing (C-HIP) Model. In Michael S. Wogalter, editor, *Handbook of warnings*, chapter 5, pages 51–61. Lawrence Erlbaum Associates, Mahwah, New Jersey, 2006.

[98] Michael S. Wogalter. Purposes and Scope of Warnings. In Michael S. Wogalter, editor, *Handbook of warnings*, chapter 1, pages 3–9. Lawrence Erlbaum Associates, Mahwah, New Jersey, 2006.

[99] Michael S. Wogalter, Todd Barlow, and Sean A. Murphy. Compliance to owner's manual warnings: influence of familiarity and the placement of a supplemental directive. *Ergonomics*, 38(6):1081–1091, 1995.

[100] Michael S. Wogalter and William Vigilante Jr. Attention switch and maintenance. In Michael S. Wogalter, editor, *Handbook of warnings*, chapter 18, pages 245–261. Lawrence Erlbaum Associates, Mahwah, New Jersey, 2006.

[101] Min Wu, Robert C. Miller, and Simson L. Garfinkel. Do security toolbars actually prevent phishing attacks? In Rebecca E. Grinter, Tom Rodden, Paul M. Aoki, Edward Cutrell, Robin Jeffries, and Gary M. Olson, editors, *CHI*, pages 601–610. ACM, 2006.

[102] Eileen Ye, Yougu Yuan, and Sean Smith. Web spoofing revisited: SSL and beyond. Technical Report TR2002-417, Dartmouth College, 2002.

[103] Zishuang Eileen Ye, Sean Smith, and Denise Anthony. Trusted paths for browsers. In *Proceedings of the 11th USENIX Security Symposium*, pages 263–279, 2002.

[104] Ka-Ping Yee. User interaction design for secure systems. In *Proceedings of the 4th International Conference on Information and Communications Security*, ICICS '02, pages 278–290, London, UK, 2002. Springer-Verlag.