OTO: Online Trust Oracle for User-Centric Trust Establishment

Tiffany Hyun-Jin Kim[†] Payas Gupta[§] Jun Han[†] Emmanuel Owusu[†]
Jason Hong[†] Adrian Perrig[†] Debin Gao[§]
[†] Carnegie Mellon University
{hyunjin,junhan,eowusu,jasonh,perrig}@cmu.edu
[§] Singapore Management University
{payas.gupta.2008,dbgao}@smu.edu.sg

ABSTRACT

Malware continues to thrive on the Internet. Besides automated mechanisms for detecting malware, we provide users with trust evidence information to enable them to make informed trust decisions. To scope the problem, we study the challenge of assisting users with judging the trustworthiness of software downloaded from the Internet.

Through expert elicitation, we deduce indicators for trust evidence, then analyze these indicators with respect to scalability and robustness. We design OTO, a system for communicating these trust evidence indicators to users, and we demonstrate through a user study the effectiveness of OTO, even with respect to IE's SmartScreen Filter (SSF). The results from the between-subjects experiment with 58 participants confirm that the OTO interface helps people make correct trust decisions compared to the SSF interface regardless of their security knowledge, education level, occupation, age, or gender.

Categories and Subject Descriptors

K.6.5 [**MANAGEMENT OF COMPUTING AND INFORMATION SYSTEMS**]: Security and Protection—*Invasive software*; H.1.2 [**MODELS AND PRINCIPLES**]: User/Machine Systems—*Human factors*

Keywords

User Interfaces for Security, Human Factors, Trust Evidence, Software Download, Trust Validation for Uncertified Software

1. INTRODUCTION

Gauging the authenticity and legitimacy of software online is challenging. For example, novice users do not understand the dangers and generally lack the ability to validate downloaded software, and even security-conscious users are often frustrated by their inability to judge the legitimacy of software.

We observe that useful trust evidence information is available on the Internet, which can help validate the correctness and trustworthiness of software. Some examples include trusted Certificate Authority-issued public-key certificates, social network-based reviews, authority or expert-based reports, PageRank information, etc. However, several factors complicate successful verification by end-users:

- **Cumbersome information gathering:** users need to spend time and effort to collect trust evidence.
- **Being aware of what evidence exists:** most users (especially non-experts) are unaware of available trust evidence indicators for validation.
- **Finding the evidence:** users may not even know where and how to start searching for evidence.
- **Assessing the quality of the evidence:** even if users find trust evidence, it may be false information (e.g., users only find evidence supporting the trustworthiness of a certain application which happens to provide malicious contents). Furthermore, contradicting evidence may exist. In this case, users may face difficulty of correctly interpreting and prioritizing the relevance of trust evidence indicators.
- **Limited trust establishment resources:** mechanisms for querying certain resources for trust establishment are currently unavailable. For example, users can rely on their online social network (OSN) friends to check if they have experience with the resources and get personal feedback; however, such a mechanism is currently unavailable.

One tempting solution would be to automate trust decisions for users based on the available evidence that systems can gather. In cases where the downloaded software is clearly malicious, such automated systems are effective, but in the frequent cases where the malware authors circumvent the automated system [15], the user is still left alone to make a trust decision. Indeed, automated protection has not been 100% accurate, due to delays in identifying malware as well as new and evolving threats.

In such a case, how can we provide useful evidence that can guide non-expert users to enhance their user experience in terms of making correct context-dependent trust decisions? To address this question, we explore how to combine multiple pieces of trust evidence and present them to users in a usable manner as follows: we furnish users with (1) an evidence list that supports the trustworthiness of the application in question, and (2) another evidence list that questions the legitimacy of the application. Applying this design to the previous example, a user downloading software would be presented with evidence showing that the software has low popularity, in which case the user would be able to weigh this evidence against the software's claim to be a commonly installed application, as is the case in many attacks.

CCS'12, October 16–18, 2012, Raleigh, North Carolina, USA.

We further explore how to customize the order of the evidence for each list based on user preferences. However, novice users may miss the critical evidence before they make trust decisions. To educate novice users, we compare the behavior between security experts and novice users and suggest what evidence experts check that novice users fail to pay close attention to.

This paper makes the following research contributions:

- We report the observation results of a user study that we conducted with security experts, to understand how they make their trust decisions when installing software. We aggregated what factors these experts take into account, which was used to inform the design of our user interface.
- We present the design of OTO, short for Online Trust Oracle, a user interface that is presented before software is installed. OTO presents two categories of evidence to assist users in determining software legitimacy: the positive evidence of why the software is safe to download, and the negative evidence for potential malware. OTO uses evidence that resists spoofing attacks.
- We present the results of a user study comparing OTO against SmartScreen Filter (SSF) on Microsoft Internet Explorer (IE) for software downloads. The results from the between-subjects experiment with 58 participants confirm that the OTO interface helps people make correct trust decisions compared to the SSF interface *regardless of their background security knowledge, education level, occupation, age, or gender*. OTO especially helps when the underlying operating system (OS) (which controls SSF and OTO) fails to correctly label the software legitimacy (i.e., the OS detects malware as legitimate, and vice versa), and when the OS correctly labels legitimate software.

2. PROBLEM DEFINITION

We address the following fundamental problem: how can we gather and present robust trust evidence indicators to assist a novice user to make a correct trust decision on a piece of software that is about to be downloaded? Numerous subproblems exist: Which trust evidence indicators are robust against adversarial influence? What trust evidence can be meaningfully and automatically gathered? How should the trust evidence be presented to the user?

Our goal is thus to design a dialog box with robust trust evidence indicators to help novice users avoid malware, even if the underlying OS fails to correctly label the legitimacy of software.

2.1 Assumptions

We assume that malware cannot interfere with OTO operation, such as the display of the OTO dialog box, the detection of software downloads, or the gathering of trust evidence. Addressing these challenges would go beyond the scope of this paper.

2.2 Desired Properties

The following properties are desired for the trust evidence indicators:

Correct. Our trust evidence information should be correct such that users can trust both the information *and* the information source. In other words, users should be able to rely on it (without searching the Internet extensively themselves) when they need to make trust decisions.

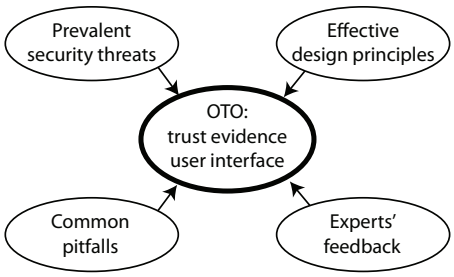


Figure 1: Factors we took into account in our design of Online Trust Oracle (OTO). We considered the most common threats and vectors in malware attacks that involve human interaction. We also drew on design principles, examined common pitfalls of novice users, and solicited feedback from experts to understand how experts make decisions regarding software installation.

When the multiple pieces of evidence result in uncertainty over whether a given piece of software is benign or malicious, users should still be able to make a correct trust decision based on how the evidence is presented.

Usable. The evidence indicators should be intuitively useful to novice users when they need to make trust decisions. Also, the information should not annoy users.

2.3 Adversary Model

Malware distributors may attempt to manipulate trust evidence information. For example, they may provide falsifying information, or hide crucial information, misguiding users to perceive their software as legitimate.

3. DESIGN OVERVIEW AND RATIONALE

Our main goal of this section is to understand the most common security threats and attack vectors involving some form of user interaction. This analysis will help us design (1) a user interface that can effectively present trust evidence and (2) user study scenarios to validate the effectiveness of the new interface. To achieve our goal, we consider four factors: analysis of popular security threats, effective design principles, common pitfalls of novice users, and feedback from experts (see Figure 1).

3.1 Analysis of Prevalent Security Threats

An important factor in designing a user interface to support trust decisions is understanding prevalent security threats. According to SophosLabs, 85% of all malware comes from the web; in particular, the top threat in 2011 was drive-by downloads where attackers lure users to malicious sites that are injected with malicious code, or to legitimate sites that host the malware [3]. According to Microsoft's Security Intelligent Report, 44.8% of successful malware attacks happen because of some action taken by the end-user [1]. This indicates that enabling users to defend themselves against online threats is critical, especially with ever increasing numbers of Internet-accessible devices, including laptops, smartphones, tablet PCs, etc.

In terms of identifying the threats, SophosLabs has identified that one of the more persistent threats of Year 2010 was fake antivirus (also known as ransomware, rogueware and scareware, mostly with a Trojan horse component) which resembles or impersonates genuine security solutions and inveigles into a victim's machine [2]. Although declining in number, possibly due to international law enforcement co-

operation, fake antivirus continued to be a big problem in 2011 [3]. Other threats include keyloggers to capture personal information and passwords, and botnet software to distribute spam, host illegal content, or serve malware.

The reports cited above list the commonly used techniques to distribute malware, which are as follows:

- *Blackhat search engine optimization (SEO)* ranks websites with malware highly in search results.
- *Social engineering click-jacking* uses social relationships to trick users into clicking on webpages.
- *Malvertising* embeds malware in advertisements displayed on legitimate, high-traffic sites.
- *Drive-by downloads* cause users to install malware simply by visiting a website.
- *Compromised legitimate websites* host embedded malware that spreads to users who access the websites.

In terms of software threats, SophosLabs and Microsoft have analyzed that malware targets Microsoft products (e.g., Microsoft office, IE, etc.), Java vulnerabilities, HTML and JavaScript, document parsers, and Adobe products (e.g., Flash, PDF) [1, 3].

All reports emphasize that educating users is one of the most critical steps to reduce the damage caused by attackers.

3.2 Common Pitfalls of Novice Users

Prior research has shown that novice users tend to encounter several common pitfalls when they make security decisions online [5, 8, 9, 18]. Such common pitfalls can help us design an effective user interface that prevents users from repeatedly making the same mistakes. We summarize the common mistakes from past work as follows:

Lack of security and systems knowledge. Users tend to misinterpret various indicators (e.g., a security lock icon, false address in the "from" line of an email, broken image, syntax of domain names, etc.) that could be used to determine whether an email or a website is trustworthy.

Visual deception. Users tend to be deceived by how an email or a website visually presents information. For example, users tend to trust a website that has professional or sophisticated layout, or that mimics the legitimate website. Moreover, users tend to be fooled by "typejacking" homograph attacks that substitute letters of a URL with similar-looking ones (e.g., `www.bankofthevvest.com`) or adding extra repeating letters (e.g., `www.gooogle.com`).

Psychological pressure. Even security experts fail to defend against messages that invoke fear, threat, excitement, or urgency [18]. For example, scareware and rogueware provide terrifying messages and force people to download (and pay for) malware. This is because such urgent cues pressure people and short circuit the available resources which in other cases help people detect deceptions.

Prior experience. Users who are familiar with particular scams seem to be good at spotting similar ones. However, this characteristic does not hold when these users are exposed to unfamiliar scams.

Bounded attention. Users tend to separate headers or URLs from the actual content of an email or a website. Users also pay insufficient attention to existing security indicators, and fail to notice when security indicators are absent.

3.3 Design Principles

In this section, we describe design principles that informed our design for interfaces that can effectively display trust evidence to users. We follow the design guideline suggested by Kuo [13] and Egelman [10] to minimize common errors in the stages of the C-HIP model [20].

Table 1: User study scenarios. To observe the mental model of security experts when they analyze the legitimacy of given software, we conducted a user study with 10 software programs, where 5 are legitimate and 5 are malicious. "Difficulty" represents how challenging we believed it would be to determine the legitimacy of a given software program.

Legitimacy	Software	Type	Difficulty
Legitimate	MindMaple	Visual organizer	Easy
	AhnLab	Antivirus	Medium
	SPAMfighter Pro	Spam filter	Medium
	Kaspersky	Antivirus	Difficult
	Rkill	Anti-malware	Difficult
Malicious	Windows activation	Ransomware	Easy
	Privacy violation	Scareware	Medium
	ActiveX codec	Trojan malware	Medium
	HDD Diagnostic	Rogueware	Difficult
	Adobe flash update	Trojan malware	Difficult

Grayed-out background. Dialog boxes on grayed-out background is shown to be effective in getting users' full attention [13]. Users also believe that such dialog boxes originate from the OS and not from malicious pop-ups. Moreover, this approach ensures that users do not overlook the dialog boxes, and forces them to make a decision [10].

Mimicked UI of OS vendor. Professional look and feel is important to enable users to develop confidence in the warning messages [13]. Hence, the warning messages should mimic the user interfaces of the OS vendor.

Detailed explanation. Users tend to ignore warnings that they do not understand [13]. Hence, it is important that the interface provides detailed explanations that people can easily understand while avoiding jargon. More specifically, dialog boxes should provide sufficient warning evidence so that users can easily understand the danger; an overview of the risks and possible consequences that users can believe and agree with; and actionable recommendations that users feel motivated to act upon [10].

Non-uniform UIs. Users tend to dismiss warnings, often due to habituation. To prevent this, it is important to provide different dialog-box designs for different levels of severity [10]. Furthermore, randomizing the order of items in dialog boxes is effective in forcing users to pay attention to security decisions [7].

3.4 Feedback from Experts

As Bravo-Lillo et al. have shown, novice users have different mental models compared to security experts [6]. In order to study how security experts determine the legitimacy of software and what advice they would give to novice users, we first ran a user study on security experts.

Based on the prevalent security threats as described in Section 3.1, we selected 10 software items for our study: 5 legitimate software programs and 5 malware programs.

Experts' user study. We created a scenario for each software in Table 1 based on the commonly used techniques for malware distribution as described in Section 3.1. We recruited 9 security experts who have been studying and working in the computer security field for at least 5 years.

Procedure. We created an interactive PowerPoint and VisualBasic mockup of the Windows 7 OS and IE browser environment to mimic a user's typical browsing behavior;

Table 2: Ten scenarios used in the experts' user study. We created each scenario based on the commonly used techniques for malware distribution as mentioned in Section 3.1. For illustration purposes, we name each expert's best friend either Alice or Bob.

Software	Scenario
MindMaple	Alice has been using ALZip, which is a free file compression program that she downloaded from the Internet. One day, Alice notices an advertisement for MindMaple on the ALZip interface and decides to click. As she clicks, a new browser window opens, showing the webpage of MindMaple – a mind-mapping software to enhance the management of multiple projects, ideas, etc. Alice skims through the webpage and she is about to click a link to try MindMaple for free.
AhnLab	Bob does not have antivirus software installed and he believes his computer is infected with a virus. Bob searches Google for 'antivirus' and navigates to a Wikipedia article that contains a list of antivirus software. After looking at several options he decides to download AhnLab V3 Antivirus. He clicks the link for AhnLab on the Wikipedia page and clicks the download link on the AhnLab website.
SPAMfighter Pro	Bob is tired of getting spam so he decides to download spam blocking software. He searches Google for 'spam filter spamfighter' and navigates to the SPAMFighter website. He clicks the 'Start Download Now' button and is redirected to a similar page with instructions to select 'Run anyway' if Internet Explorer SmartScreen issues a warning. He clicks on 'Start Download Now.'
Kaspersky	Alice's computer does not have an antivirus program installed, so she decides to install one. She searches Google for 'antivirus software' and decides to click on the sponsored link for Kaspersky. She clicks on the link for a free 30-day trial, fills in some personal details, and clicks the download button.
Rkill	Bob believes his computer is infected with malware. He reads two forums on free malware removal software and an article describing Rkill as the repair tool of the week. He decides to try Rkill and searches Google for 'rkill download.' Bob clicks on the link for the CNET download page and then clicks on the 'Download Now' button.
Windows Activation	Alice's PC was installed with a pirated copy of Windows 7 which she downloaded from a file sharing website. She has had no problems with her PC so far, and is completing a homework assignment that requires a photo editor. Alice searches Google for 'free adobe photo editor torrent' and selects the first link in the search results. She is trying to find the download button when a window appears prompting her to call a number to receive an activation code for Windows.
Privacy Violation	Alice has noticed that her PC has been running slowly, so she searches Google for 'windows 7 is slow.' After looking through several options she navigates to a page titled 'Registry Clean Up.' While browsing the webpage a window appears with a warning about privacy violations found on her computer. She clicks on the 'Repair Now!' button.
ActiveX Codec	Bob is bored at home and decides to watch a movie. He searches Google for 'batman begins' and selects a UStream video. While waiting for the video to load a dialog box appears over the video window stating that there was an error displaying the video and prompts him to install ActiveX to fix this error.
HDD Diagnostic	Alice's computer has been making some weird noises. Alice decides to search for a solution on Google. She finds a webpage titled 'Windows 7 making weird buzzing noises.' As she browses the webpage, a window appears claiming that it is scanning the computer to analyze PC performance. Alice clicks the 'Pause' button.
Adobe Flash Update	Bob enjoys watching videos by 'Just for laughs TV.' He searches Google for 'just for laughs gags' and looks through several options before selecting a Youtube video to watch. After Bob finishes the first video, he decides to click on a related video. While waiting for the video to load a dialog box appears over the video window prompting him to download an update for Adobe Flash Player.

each scenario is composed of a sequence of Windows 7 desktop screenshot slides, each of which shows a web browser with a different website based on the scenario. The next slide is triggered when a user clicks an appropriate area inside the browser. For example, the ActiveX codec scenario starts with a Google result page of 'batman begins,' and asks the user to click a particular link (e.g., UStream link) to proceed. Such a click triggers the next slide, displaying the UStream webpage with an active progress bar. Shortly after, the following slide displays an ActiveX codec error, asking the user to install ActiveX to fix this error.[1]

To obtain responses that experts would provide in typical situations, we asked each expert to pretend to be watching a close friend who is using a computer and is about to download software. For this role-playing, one of the authors pretended to be a close friend of each expert, and we asked each expert to observe the author's browsing behavior for 10 different scenarios. For each expert, we presented 10 scenarios in randomized order. Table 2 summarizes the scenarios that we used for our study.

At the end of each scenario, we asked the following questions to analyze the types of evidence that experts use to determine the software's legitimacy:

- Would you recommend that Alice proceeds and downloads the software? [Yes / No / Not sure]
  - **[If Yes or No]** Why?
  - **[If Not sure]** What would you do to find out the legitimacy of this software?
- What evidence would you present to Alice to convince her of the legitimacy of this software?
- How well do you know this software? [1 (don't know at all) – 5 (know very well)]

The last question is to analyze whether their prior knowledge affects how they recommend software. After all 10 scenarios, we asked each expert to draw a flow chart that shows how (s)he decides about the software legitimacy in general. We emphasized that this flow chart is to help educate their close friends in determining the software legitimacy. At the end of the study, we asked each expert to provide feedback on the download dialog box that we designed.

Results. In general, the first action that all the experts took was examining the downloading software's *hosting website* carefully, regardless of their prior knowledge. All 9 experts emphasized the importance of downloading software from a trusted and reputable website, and indicated that they would go directly to the publisher of the software. If users are not on the publisher's website, the experts would ascertain if the URLs and the website content looks legitimate. If the hosting website does not look reputable, the

[1]Note that our mockup of the Windows 7 environment is indistinguishable to the real Windows 7 environment, as commented by study participants.

Table 3: Summary of 9 experts' processing operations. This table illustrates processing operations that the experts indicated in their own flowchart, and the number of experts who mentioned the same processing operation.

Processing operation	#
Software review	
• Are reviews available from reputable sources, experts, or friends?	9
• Are the reviews good?	3
• Do a lot of people use the software?	1
Hosting site	
• Is the hosting site reputable?	8
• What is the corporate parameter (e.g., number of employees, age of the company) of the site?	2
• Is the site related to the software to be downloaded?	1
• Are you being redirected from advertisements, emails, etc.?	1
• Does the site have a high rank on Google using general search terms?	1
• Does the company have a physical location near where you live?	1
User intention	
• Did you search for that specific software?	1
• Are you familiar with this software?	1
• Do you really need it?	1
• Are you downloading from a pop-up?	1
Securing machines	
• Do you run an updated antivirus?	2
• Is your machine trusted?	1

experts would search for the same software from a reputable source. Alternatively, 8 experts agreed that they would consider the (unknown) hosting site as reliable if Google's search query with *general* terms (e.g., "antivirus") provides a high page rank for that host; on the other hand, 1 expert mentioned that it would be "possible to poison Google's search results" for very specific terms (e.g., "AhnLab V3 antivirus").

All the experts also emphasized *user intention* to be an important factor for judging software legitimacy. For example, all experts pointed out, throughout the study, that they would not recommend their friends to download software from pop-up windows conveying information that is irrelevant to the original search. This is because the software advertised in the pop-up window is highly unlikely to be what the user intends to download, and highly unlikely to be legitimate. On the other hand, if users proactively click to download the software that they have been searching for, the experts mentioned that the likelihood of downloading malware may be low.

All the experts also emphasized that carefully examining *software reviews by trusted authorities or experts* is critical. Before downloading software, all the experts would recommend users to research the downloading software's reviews from reputable sources (besides the hosting site). For example, the experts suggested checking (1) the total number of people currently using the software, (2) positive and negative reviews, and (3) business information for the software vendor.

Two experts mentioned that *securing machines* is also important. For example, one way to reduce malware download would be running antivirus software that is up-to-date.

Flowchart. At the end of the study, we asked each expert to draw a flowchart that their friends could use as a guideline for future downloads. In general, every expert's detailed thinking process was unique. However, they all illustrated the same main points, in particular *hosting site, user intention, software review, and securing machine* as described above. Table 3 summarizes the frequency of different process operations that the experts drew. We merged all

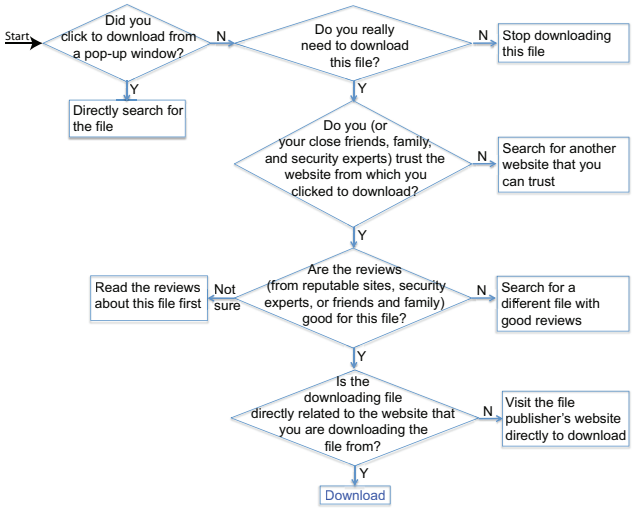


Figure 2: A merged and simplified flowchart. This flowchart preserves the order of the processing operations as the majority of the experts indicated.

experts' flowcharts to guide our dialog-box design, with an additional processing operation: Were you urged to download? This is because many antivirus malware programs, especially ransomware, scare users into immediately downloading the program to recover their computers. Figure 2 shows the resulting flowchart.

Discussion. An interesting observation from this study was that the experts did not rely on special knowledge and their reasoning was simple. Although this study did not reveal groundbreaking facts from security experts, such results are promising since an enhanced user interface may empower even novice users to make correct security decisions.

4. OTO: USER INTERFACE WITH TRUST EVIDENCE

In this section, we introduce OTO, which we designed and modified based on the four design points and the results/feedback from the expert study in Section 3. OTO is an interface that displays clues about the safety of downloading files.

OTO provides both reasons why the downloading file may be safe to install and reasons why the downloading file may be harmful to the computer. Figures 3–5 show examples of the OTO interface. With OTO, users can make informed decisions about proceeding with or canceling the downloads. In Section 4.1, we describe the design overview of OTO, and in Section 4.2, we delineate a list of evidence types that are suitable for OTO. In Section 4.3, we describe how evidence is displayed in the OTO interface.

4.1 OTO Interface Overview

As shown in Figure 3, OTO is displayed when a user's action leads to the downloading of software; for example, the user actively clicks to download or (s)he clicks the "cancel" or "close" button but the download continues.

We assume that a system-level trusted path exists, such that when a file is about to be downloaded, the OS can detect it and activate OTO. When OTO is activated, the entire screen grays out, which will catch the users' full attention and ensure that OTO is a legitimate OS-certified program and not another malware program.

Figure 3: An example of an OTO interface. OTO presents reasons why the file may be harmful and why the file may be safe to install.

Figure 4: An example of an OTO interface for legitimate software. This example presents 0 negative reason and 8 positive reasons.

Based on how OTO judges the software's legitimacy, OTO shows (1) reasons why this software is safe to download (positive evidence), and (2) reasons why this software may harm the computer (negative evidence). Note that these reasons are primarily based on the security experts' suggestions that can be feasibly gathered from the online sources. Users can also choose to see other reasons that are helpful in determining the downloading software's legitimacy. We also envision that a user can rank how important and critical each piece of evidence is to him, and OTO will assess the software's safety level based on such user-driven evidence ranks.

Following the Windows User Account Control framework, OTO has three color modes for different safety levels:

Blue with "?" shield symbol for legitimate software. As shown in Figure 4, this applies to software that is highly likely to be legitimate. For example, software that is digitally signed by Microsoft would be categorized to show the blue OTO interface. This interface shows the following warning statement against a blue background: "Are you sure you want to download this file? It may damage your computer. We gathered evidence to help you decide." Next to the statement, Microsoft's blue security shield symbol with "?" is displayed. Under this blue message box, a table

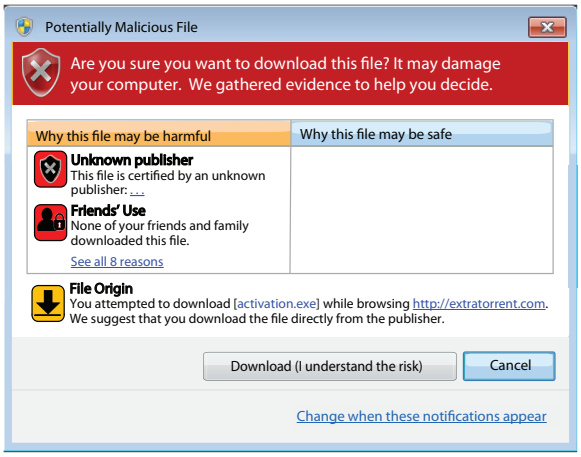


Figure 5: An example of an OTO interface for malware. This example presents 8 negative reasons and 0 positive reason.

is displayed, showing negative and positive reasons. For the software in this category, a significant number of positive pieces of evidence is displayed, although some negative evidence may be displayed (if it exists). As shown in Figure 4, OTO shows the top two pieces of evidence, which either OTO or the user has selected as important, and contains a link that users can click if they want to see all 8 reasons.

Red with "X" shield symbol for malicious software. As shown in Figure 5, the red colored OTO interface is for pieces of software that is highly likely to be malicious. As with the blue OTO interface, the same warning statement is shown; however, the statement is against a red background, with a red "X" security shield symbol. Also, this interface will display a significant number of negative pieces of evidence, although few positive pieces of evidence may be displayed.

Yellow with "!" shield symbol for software of uncertain provenance. As shown in Figure 3, the yellow colored OTO interface is for software that the system cannot clearly determine as either malicious or safe. As a result, with a yellow "!" security shield symbol, the same warning statement is shown against a yellow background. For this software, the interface will display both positive and negative evidence.

4.2 Retrieving Positive and Negative Evidence

We conjecture that users fail to make informed trust decisions due to (1) unawareness of existing evidence (e.g., they are simply unaware of the ways to find evidence or are reluctant to do so), (2) overabundance of evidence (e.g., too much, and potentially conflicting evidence further confuses them), and (3) the cumbersome nature of searching for evidence. Understanding these issues are part of the objectives of this paper, and providing a concise list of useful, helpful, and objective pieces of trust evidence may enable users to make informed trust decisions.

Existing tools and technology websites provide some evidence that people can refer to. For example, CNET[2] provides reviews and specifications about software; specifications include average user rating, the software's version number, file size, release date, number of total downloads, number of downloads last week, etc. Norton also has antivirus tools featuring File Insight, the purpose of which is to pro-

[2] http://www.cnet.com

Table 4: A list of trust evidence that can be provided to users when they download software. Each piece of evidence is scalable for automatic retrieval from the Internet; this table presents which pieces of evidence that can be made robust against spoofing attacks.

Evidence	Robust
General information about software	
• Developer	×
• Version Number	×
• Date added	×
• File size	×
• File name	×
• Category of software (e.g., video player, game)	×
• Digitally-signed certificate	✓
Origin of the downloading software & developer	
• Hosting site (that users click to download files)	✓
• Source address (where files are downloaded from)	✓
• PageRank of hosting site and source address	✓
• Information about the developer (e.g., business history, location, number of employees, etc.)	×
Crowdsourcing	
• Total number of people who downloaded the software	✓
• Total number of people who downloaded last week	✓
• Ratings by people who downloaded the software	✓
• Reviews	✓
• Recent changes in ratings	✓
Reputable 3^{rd}-party resource	
• (OSN) Expert friends who also downloaded the same software	✓
• (OSN) Suggestion of more popular/credible software within the same category	✓
• (Authorities) Credible newspaper/magazine articles featuring the software	✓

vide specifications about the files that users are about to download. Similar to CNET, Norton File Insight provides the information regarding developers, release date, software's version number, number of Norton community users who have used the same software, date when the software was released, Norton's rating on the software, where the file is being downloaded from, etc.

However, some of the specifications provided by the existing tools and technologies may not be robust against spoofing attacks. For example, attackers can easily increase version numbers to seem as though the software providers update the software frequently. Similarly, the release date of the software could potentially be backdated.

As a first step, we have analyzed the pieces of trust evidence that are both *robust* against spoofing attacks and *scalable* for automatic retrieval from the Internet. Table 4 lists the evidence that satisfies our criteria.

Using the evidence that is both robust and scalable, we describe how OTO displays them in the next section.

4.3 Presenting Positive and Negative Evidence

Each of the pieces of trust evidence, as shown in Table 4, can be perceived as positive (supporting that a file may be safe to download) or negative (confirming that a file may be harmful). For example, "Friend's Use" can be a negative reason if none of a user's security-expert friends and family downloaded the same file (Figures 3 and 5), but it can be a positive reason if many friends and family downloaded the file (Figure 4).

For the evidence from authorities, we assume that OTO uses an algorithm that can determine if an article is positive on a piece of software or whether it declares the software as malware. Given the recent advances in text understanding, we assume such analysis to be viable and reliable [4].

Note that the OTO interface displays a summary of each evidence type. If users want to examine the evidence in detail, OTO provides a clickable link that will display more detailed explanation regarding that evidence.

The OTO interface only displays 2 pieces of supporting evidence for each column. If there are more than 2 pieces of evidence, OTO provides a clickable link that can show all supporting evidence for the particular column.

Furthermore, OTO considers the "File Origin" evidence as neutral evidence because determining whether the downloading file is legitimate, based on the source address, is a non-trivial question. As a result, "File Origin" is displayed below the reasoning table. If OTO is able to consult credible blacklists of malicious websites, OTO can alert users if the hosting site was found on the blacklist(s).

5. SECURITY ANALYSIS

In this section, we delineate how OTO detects malware (Section 5.1) and how OTO handles misclassifications of software (Section 5.2). In Section 5.3, we also describe how OTO mitigates manipulation attacks as mentioned in Section 2.3.

5.1 Detecting Malware

In this section, we describe how OTO successfully detects malware and provides helpful evidence to users. In general, two types of malware are as follows:

Zero-day malware. In case of zero-day malware, lack of available positive evidence should alarm users to be careful about downloading the file.

Well-known malware. If the malware is well-known, the evidence gathered by the OS is likely to contain substantially more pieces of negative evidence than positive evidence. Given a high number of negative evidence and the strength of negative evidence (compared to those of positive evidence), there is a high probability that users will detect such malware as harmful.

5.2 Handling False Alarms

There is a possibility that OTO misclassifies software: (false negative) given undeniably malicious software, OTO may fail to detect it as malicious when the number and/or the quality of positive evidence is proportional to that of negative evidence; (false positive) OTO may misclassify legitimate software as malicious.

In such cases, users can detect false alarms by examining the evidence. For example, as shown in Figure 3, Alice can analyze each piece of evidence and judge whether negative reasons outweigh positive reasons: are (1) a threat blog discussion on this software and (2) no usage by the user's security-expert friends and family (along with a third reason) stronger than (1) high number of available reviews, (2) high popularity, and 2 other reasons? Since all pieces of evidence are drawn from Alice's preferences to help her decide, Alice can make her own informed trust decision based on the displayed information.

One may argue that Alice may still find it difficult to analyze the displayed information. However, what is displayed is likely to be what she would have searched online herself. Given the easy availability of the information increases the likelihood that she will consider the information. Plus, the evidence originates from trustworthy resources that she defines. Hence, she is safe from reading misguiding information that she could have found on the Internet, and OTO introduces no extra risk to Alice.

5.3 Mitigating Manipulation Attacks

The OTO system is designed to disrupt malware distributors from manipulating positive and negative evidence. We discuss three potential categories of attack.

Generating falsifying evidence. Attackers can attempt to create fake positive evidence for malware. In Table 4, we present the list of pieces of evidence that we believe can be made robust. For example, the total number of downloads can be made robust by analyzing the user population and temporal aspects of the download – in case the attacker rents a botnet to inflate the number of downloads the sudden spike in downloads and the population distribution should appear anomalous.

While it is outside the scope of this paper to ensure the robustness of each indicator, we would like to emphasize that a successful attack would need to forge several positive pieces of evidence, which will likely be negated by a strong negative piece of evidence that would still alert the user.

If an attacker creates positive articles about the software, it would need to get them published at an authoritative site trusted by the user to be considered by OTO.

Hiding harmful evidence. Attackers may attempt to prevent the OS from fetching negative evidence or prevent an online resource from serving negative evidence. While these attacks are outside the scope of this paper, we believe that it would be challenging for attackers to prevent authoritative resources from serving information about the malware.

Impersonation of legitimate software. Attackers may attempt to impersonate a well-known software system and claim an update or a free installation. We assume that OTO can perform secure associations of the pieces of evidence with the software, for example using a cryptographic hash value of the software. It will, however, be challenging for written articles to form the correct association with the correct piece of software, as journalists rarely include the cryptographic hash of the software they are writing about. In case OTO is widely deployed, mechanisms for associating pieces of evidence with software need to be used, for example by including the public key of the software distributor or the cryptographic hash of the software in a written article.

6. EVALUATION

We conducted a user study to test whether OTO achieves the desired properties as described in Section 2.2.

6.1 Demographics

We recruited 58 participants from a diverse set by advertising on Craigslist, flyers around bus stops, Facebook postings, and university mailing lists. We randomly assigned 29 participants to each of the OTO and SSF conditions. Table 5 summarizes the demographics.

6.2 User Study Process

We designed a between-subjects experiment with two conditions, the SSF and OTO conditions, each of which consists of 10 scenarios as described in Table 2.

We consider the SSF condition as a baseline for the following reason: According to NSS Labs, SSF is the current state-of-the-art technology[3] that is widely used on a modern browser, which ensures that the users are well aware of IE's interface from prior experience. SSF checks the software that a user clicks to download against a known blacklist of malicious software. If the software is flagged, then a red-banner warning appears, intentionally delaying the download procedure, and the user must click on the "Actions" button to proceed with the download. Figure 6 depicts the two SSF status bars that appear at the bottom of a browser: a warning and a normal download deemed legitimate.

[3] `http://www.pcmag.com/article2/0,2817,2391164,00.asp`

Table 5: Demographics of study participants for OTO and SSF conditions.

		OTO	SSF	Total
Group size (N)		29	29	58
Gender	Male	15	15	30
	Female	14	14	28
Age	Minimum	20	18	18
	Maximum	48	59	59
	Average (μ)	26.7	29.0	27.8
	Std dev (σ)	6.5	10.4	8.7
Occupation	Student	16	16	32
	Other	13	13	26
Educational background	High school	3	8	11
	Bachelor	19	14	33
	Post-graduate	7	7	14
Security knowledge level	Novice	2	4	6
	Intermediate	19	17	36
	Expert	8	8	16

For our experiment, we modified the mockup slides for the experts' user study in Section 3.4 as follows:[4] two conditions differ in the last step of each scenario when the interviewer, who is role-playing as a close friend of the participant, clicks on the download button. The SSF condition prompts IE's SSF dialog box (Figure 6), while OTO prompts the evidence interface (Figures 3–5). Figures 7 and 8 show the snapshots of the SSF and OTO interfaces, respectively. Similar to the expert user study as described in Section 3.4, we asked each participant to pretend that (s)he was helping a close friend make a trust decision when downloading a piece of software from the Internet.[5]

Pre-study. Before we started the experiment, we presented 6 pre-study questions in random order to identify each participant's security knowledge level (see Table 6 for questions). We then classified the participants as novice, intermediate, or expert if they correctly answered 0–2, 3–4, or 5–6 questions, respectively. Table 5 shows the classification of the participants based on their security knowledge.

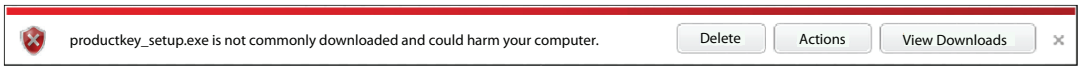
During study. Before we began, we told the participants that they would be given 10 scenarios, where some software may be safe to download and some may be harmful to the computer. Their task would be to recommend whether the friend should proceed or cancel the download. We also explained that each participant would be paid $15, plus an additional $1 for each question that they answered correctly, with a maximum payment of $25. This payment method was to incentivize the participants to provide the best educated guess and mimic reality, rather than be extremely-conservative or liberal.

[4] From the end-user's point of view, only user experience matters, and participants commented that our study was indistinguishable from a real environment.

[5] We observed that participants cared more about providing honest/correct advice to close friends than about protecting the laptop of an unrelated person (i.e., the interviewer). Hence, the role-playing increased participation through passive help. Furthermore, our experiment setting ensured that all participants were tested using the same browser and OS settings.

Table 6: A pre-study questionnaire to identify the participants' security knowledge.

- While browsing the web, you see a pop-up window showing that your computer is infected with viruses. This warning recommends that you download antivirus software to delete the viruses from your computer. Do you think this software is safe to download?
- One of your close friends recommend a product called K7 SECURE-WEB for secure online financial transactions. You google the product and click the #1 top search result, which redirects you to `http://www.k7computing.com/en/Product/k7-secureweb.php`. You see a link to purchase this product for $19.99. Do you think this software is safe to purchase?
- You log into Facebook and see that your friend posted a video clip entitled: OMG I just hate RIHANNA after watching this video. When you click the video, you get a prompt asking you to install ActiveX codec player to watch this video clip. Do you think this video codec is safe to download?
- While browsing the web, you see an advertisement about Google's super-fast Chrome browser. When you click this advertisement, you see an instruction page to install Chrome from `https://www.google.com/chrome/`. Do you think this software is safe to download?
- You receive an instant message, such as AOL IM, MSN messenger, Google talk, etc., from your friend to download identity protection software. When you click the link, it redirects you to `http://download.cnet.com/` to download the software. Do you think this software is safe to download?
- You are checking your email and you see that you have an urgent message from your bank. The email message says that the bank is upgrading their software to improve your safety and security, and asks you to install the software that is attached to the email. Do you think this software is safe to download?



(a) A SSF warning for potentially malicious file download.

(b) A SSF interface for normal downloads.

Figure 6: Microsoft IE's SmartScreen Filter (SSF) interfaces.

Figure 7: A screenshot of a SmartScreen Filter (SSF) interface. The SSF dialog box is shown at the bottom of the IE browser.

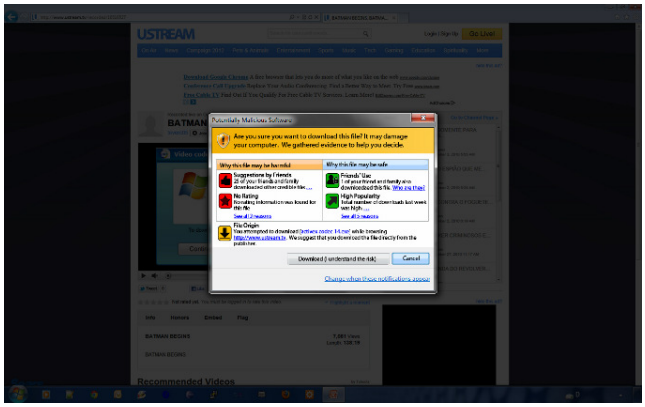


Figure 8: A screenshot of an OTO interface. The OTO dialog box displays positive and negative pieces of evidence on the grayed-out screen.

During the experiment, the interviewer walked through the scenarios, and asked the same questions that we used for the expert user study in Section 3.4. For this experiment, we categorized the ten scenarios into the following four cases:

1. **True positive (TP)**: the system *properly* detects malware as malicious. In this case, SSF displays a red warning bar on the bottom of the screen, and OTO displays a red interface with only negative evidence.
2. **True negative (TN)**: the system *properly* detects the legitimate software as legitimate. In this case, SSF displays a yellow warning bar on the bottom of the screen, and OTO displays a blue interface with only positive evidence.
3. **False positive (FP)**: the system *incorrectly* detects the legitimate software as malicious. In this case, SSF displays a red warning bar on the bottom of the screen, forcing users to stop downloading, and OTO presents a yellow OTO interface displaying more suspicious evidence than trustworthy evidence.
4. **False negative (FN)**: the system *incorrectly* detects malware as legitimate. In this case, SSF displays a yellow download warning bar on the bottom of the screen, and OTO displays a yellow OTO interface with more positive evidence than negative evidence.

Among 10 scenarios, we assigned two pieces of software with easy to medium difficulty level to TP and TN cases, and three pieces of software with medium to difficult level to FP and FN cases. Table 7 describes the scenario assignments.

When the interface prompted, we asked each participant if (s)he thought the software was safe to download or harmful

Table 7: Scenario assignments for TP, TN, FP, and FN cases.

Ground truth	System detection outcome: Legitimate	System detection outcome: Malicious
Legitimate	TN Ahnlab MindMaple	FP Rkill Kaspersky SPAMfighter
Malicious	FN ActiveX codec HDD diagnostic Adobe flash	TP Windows activation Privacy violation

to the friend's computer. Participants were given the following answer choices: (1) legitimate, (2) malicious, and (3) unsure (each scenario had 1 correct answer and "unsure" was counted as incorrect). We asked them to think aloud, and the PowerPoint automatically logged the time duration until the participants made trust decisions. After they decided, we asked for justifications and while the users were thinking aloud, the interviewer logged the answers for analysis.

Post-study. After the experiment, we asked the participants to answer questions to obtain feedback on the interfaces. More specifically, we asked how much the different types of evidence (as shown in Table 4) would help the participants when downloading software.

6.3 General Observations

In this experiment, we observed that many participants tend to trust their past experience and security knowledge in certain scenarios that they are familiar with (e.g., TP and FN scenarios); hence they used the SSF and OTO interfaces for confirmation of their judgments.

In the SSF condition, many participants were conservative when prompted with a pop-up that they did not initiate. In the OTO study, however, we noticed that the participants began to rely on the evidence that the OTO interface provided, especially for those scenarios that were unfamiliar to them. Many participants claimed that the websites looked legitimate but they were not aware that the websites existed. This allowed them to focus their attention on the evidence to gain more knowledge, especially for TN and FP scenarios.

One concern with the OTO interface was that users may need to spend time to read the evidence. However, we observed that the participants did not take a significantly longer time with the OTO interface, as compared to the SSF interface. Figure 9 shows the average response time from 24 random participants ($N = 11$ for OTO, $N = 13$ for SSF) who did not think aloud while making trust decisions, and Table 8 summarizes the average and maximum time. One interesting observation is that overall, the participants took less time to make trust decisions for OTO compared to SSF. More specifically, the participants took approximately the same amount of time for FN cases regardless of the interface they were given, and the participants given the OTO interface took *less* time to decide for FP, TP, and TN cases. This result is due to the following observation from the experiment: In the SSF condition, many participants took some time to make trust decisions given scenarios that they never experienced before. In the OTO condition, however, the participants relied on the displayed evidence lists to make trust decisions, resulting in faster decisions compared to SSF.

Next, we analyze in detail the effectiveness of OTO as compared to SSF.

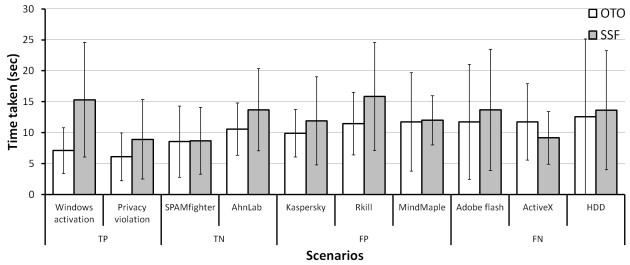


Figure 9: The average time that the participants spent to make a trust decision on each scenario for SSF and OTO conditions ($N = 13$ for SSF, $N = 11$ for OTO).

Table 8: Mean and maximum time that the participants spent to make trust decisions ($N = 13$ for SSF, $N = 11$ for OTO).

	OTO		SSF	
	Mean	Max	Mean	Max
Overall	10.1 ± 6.9	48	12.3 ± 7.6	34
TP	6.6 ± 3.7	14	12.1 ± 8.5	32
TN	9.5 ± 5.0	20	11.2 ± 6.5	26
FP	11.0 ± 5.8	31	13.3 ± 7.0	31
FN	12.0 ± 9.4	48	12.2 ± 8.4	34

6.4 Effectiveness Analysis

We analyze if the OTO interface helps users make correct trust decisions compared to Microsoft's SSF interface (base case). Based on the number of correct answers provided by 58 participants (29 participants for each condition), we ran a Repeated Measures ANOVA test. The results confirm that the OTO interface helps people make more correct trust decisions compared to the SSF interface *regardless of the participants' background security knowledge, education level, occupation, age, or gender.* Table 9 summarizes the results.

We designed our study to test whether OTO helps users make correct trust decisions even if the OS mistakenly categorizes legitimate software as malicious and vice versa. Below is a list of hypotheses for 4 cases, along with detailed analysis on how our study participants responded for 4 cases using Mixed Models. In general, a significant main effect exists for different interface conditions with 4 cases taken into account ($F(1, 65) = 18.1$, $p < .001$), and the main effect of 4 different cases is significant ($F(3, 517) = 12.61$, $p < .0001$). Table 9 summarizes the results for specific cases.

True Positive (TP). In the TP case, the OS correctly identifies malware. We tested two TP scenarios ("Windows activation" and "Privacy violation"), given the following hypothesis:

HYPOTHESIS 1. *When software is malicious and the OS detects it as malicious, users given the OTO interface detect the malware at least as well as those with the SSF interface.*

The Mixed Model delivered no significant difference between OTO and SSF for TP scenarios. Hence, Hypothesis 1 is valid and OTO performs at least as well as SSF for the TP case.

True Negative (TN). In the TN case, the OS correctly identifies legitimate software as legitimate. For 2 TN scenarios, namely "AhnLab" and "SPAMfighter Pro," our hypothesis is as follows:

HYPOTHESIS 2. *When software is indeed legitimate and the OS detects it as legitimate, users given the OTO interface detect the legitimacy at least as well as those given the SSF interface.*

Table 9: Summary of Repeated Measures ANOVA (Overall), Mixed Models (TP, TN, FP, FN), and ANOVA (Usefulness, Annoyance) results for the effectiveness of OTO compared to SSF ($N = 58$, 29 for each condition). The higher mean that is statistically significant from the other is highlighted in bold.

	Overall min: 0, max: 1		TP min: 0, max: 1		TN min: 0, max: 1		FP min: 0, max: 1		FN min: 0, max: 1		Usefulness min:1, max:5		Annoyance min:1, max:5	
	μ	$\sigma_{\bar{X}}$	μ	$\sigma_{\bar{X}}$	μ	$\sigma_{\bar{X}}$	μ	$\sigma_{\bar{X}}$	μ	$\sigma_{\bar{X}}$	μ	σ	μ	σ
OTO	**.86**	**.03**	.96	.05	**.91**	**.05**	**.79**	**.05**	**.79**	**.05**	4.24	.58	**3.75**	**1.02**
SSF	.67	.03	.93	.05	.64	.05	.60	.05	.57	.05	4.10	.72	3.17	1.10
Results	$F(1,51) = 4.03$ $p < .0001$		$F(1,425) = .18$ $p = .67$		$F(1,425) = 12.49$ $p = .0005$		$F(1,294) = 8.86$ $p = .003$		$F(1,294) = 11.09$ $p = .001$		$F(1,57) = .64$ $p = .43$		$F(1,57) = 4.40$ $p = .04$	

μ: mean, $\sigma_{\bar{X}}$: standard error, σ: standard deviation

There was a statistically significant difference between OTO and SSF conditions for TN scenarios, as shown in Table 9. Hence, Hypothesis 2 is valid and users do indeed make significantly better trust decisions with OTO compared to SSF.

False Positive (FP). In the FP case the OS identifies legitimate software as malicious. We tested three FP scenarios ("Kaspersky", "Rkill", and "MindMaple") given the following hypothesis:

HYPOTHESIS 3. *Compared to SSF, OTO enables users to download legitimate software that the OS mistakenly detects as malicious.*

There was a statistically significant difference between OTO and SSF for FP scenarios, as shown in Table 9. Hence, for cases where browsers miscategorize legitimate software as malicious, OTO's trustworthy and suspicious evidence significantly helps users make correct and informed trust decisions.

False Negative (FN). In the FN case, the OS fails to detect malware. For three FN scenarios ("HDD Diagnostics", "ActiveX codec", and "Adobe flash update"), our hypothesis is the following:

HYPOTHESIS 4. *Compared to SSF, OTO prevents users from downloading malware that the OS does not detect.*

We found a statistically significant difference between OTO and SSF for FN scenarios (see Table 9). Hence, for cases in which browsers fail to detect malware, the trustworthy and suspicious evidence provided in the OTO interface significantly helps users make correct trust decisions.

Discussion. The marginal interaction effect ($F(3,517) = 2.04$, $p = .10$) suggests that the TN case was especially helped by OTO. It also shows that the FN and FP cases were helped by OTO, but not the TP case. Hence, OTO's approach of providing positive and negative trust evidence helps people make *correct* trust decisions, even if the interface misclassifies software.

6.5 Usability Analysis

Usefulness. During the post-test of both SSF and OTO studies, we posed a 5-point Likert scale question to measure how useful participants found the corresponding interface, with the following hypothesis:

HYPOTHESIS 5. *Users find the OTO interface at least as useful as the SSF interface.*

There was no statistically significant difference between two interfaces for usefulness (see Table 9), hence satisfying our hypothesis.

Annoyance. We asked participants how much they were annoyed by given interface. Our hypothesis was as follows:

HYPOTHESIS 6. *Since the OTO interface can potentially contain more information than the SSF interface, users may find the OTO interface more annoying than the SSF interface.*

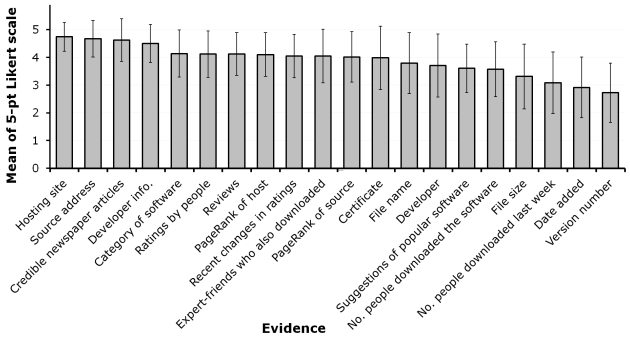


Figure 10: Mean and standard deviation of 20 forms of trust evidence using 5-point Likert scales. Participants rated how they consider each evidence as helpful in validating the software legitimacy on a 5-point Likert scale.

We were able to find a statistically significant difference between two interfaces, nullifying Hypothesis 6. In other words, the participants found OTO more comfortable to use.

6.6 Desired List of Evidence

At the end of each study, we gave participants a list of pieces of evidence for assessing software legitimacy (same as what is presented in Table 4), and asked them to rank the usefulness of each piece of evidence on a 5-point Likert scale. Figure 10 shows the mean and the standard deviation of each piece of evidence.

In general, participants reported the robust and scalable evidence as helpful for trust assessments, except information about the developer and the category of software. In particular, participants found the evidence suggested by security experts helpful.

7. RELATED WORK

In this section we discuss related research covering various aspects of security warning design and online user behavior: user mental models of security risks, mitigating the effect of habituation, and user assessments of content credibility.

7.1 User Mental Models

Prior research has demonstrated the importance of a user's mental model of security risks. Sunshine et al. have analyzed how risk perception plays a large role in how people respond to SSL warning messages, which are used to notify users of a potential Man-in-the-Middle attack [17]. Their results show that, in general, all the warning signs did not prevent many of the users from exhibiting unsafe behavior, even for security experts who perceived the warnings as a lower risk than, for example, a mismatched certificate. This may be an even bigger issue for those not familiar with computing security and privacy because their perception of risk may

be even lower. For example, these users may be unaware of the incentives for cyber-criminals or the scalable nature of online attacks and therefore do not perceive a significant personal risk.

Similarly, Bravo-Lillo et al. have addressed psychological responses to warning messages in terms of how users perceive and react to computer alerts [6]. Using various warnings from popular operating systems, their study revealed that many users have the "wrong" mental model for many computer warnings. For example, understanding SSL warnings is difficult if one does not know about certificates or Man-in-the-Middle attacks. The authors argue that warnings should be a third line of defense, after designing out the risk and guarding against the risk.

Wash illustrates that home users tend to follow some (but not all) pieces of security advice from experts based on the identification of folk models of security threats [19]. The author finds that some users with certain folk models (e.g., viruses are generally bad, viruses are buggy software) believe that the avoidance of intentional downloading and execution is enough protect them from virus infection. Based on the analysis of eight folk models on viruses, hackers, and botnets, the author suggests that security technologies should not only focus on actionable advice, but also clearly explain potential threats that users may face.

Motiee has explored how to create informative contents to Microsoft Windows 7 User Account Control warnings [14]. More specifically, the author focuses on the information content to help users assess risk and correctly respond to warnings. Based on the user study, the author states that the most understandable and useful pieces of content for users are as follows: program name, origin, description, certification, changes to apply and result of antivirus scan. The author also suggests selecting a context-based subset of the content to avoid habituation. However, the majority of the contents is not robust against spoofing attacks, thus questioning the effectiveness of the warnings.

We address the requirement of a correct mental model for security risk assessments by incorporating robust traditional evidence (e.g., file origin information) with evidence that may be more familiar to novice computer users (e.g., OSN data and reviews from authorities) and presenting the information in an intuitive way such that users can make informed trust decisions without heavily relying on the security assessment by the underlying operating system.

7.2 Habituation

The research community has studied the effects of habituation of users to warning dialogs. Egelman et al. have analyzed the effectiveness of browser warnings [11]. Specifically, they compare the effectiveness of passive and active warnings in getting users to avoid spear phishing attacks. They found that the majority of users (97%) were deceived by least one of the phishing messages. Many users simply did not notice warning signs. However, 79% of the participants presented with active warnings heeded them. This result confirms that habituation is a major road block for creating effective warning messages.

To mitigate the tendency of users who ignore security dialogs, Brustoloni et al. have explored the use of polymorphic and audited dialogs to defend against risky email attachments [7]. Since a polymorphic dialog continually changes, users are forced to examine the dialog even if their goal is to bypass it as quickly as possible. Auditing involves warning users that their responses will be forwarded to an auditor and their account may be quarantined based on those responses. Their results show that untrained users accept significantly less unjustified risk using polymorphic and audited dialogs as compared to conventional dialogs, but at the cost of usability: Expert users who already know what decisions they want to make may find a constantly changing interface annoying. Furthermore, it may be challenging to enforce an auditing policy in cases where there is no central authority (e.g., home or public network).

OTO mitigates habituation by varying the color of warning dialog boxes based on the severity of the risks. Moreover, every software download generates a unique set of evidence, encouraging users to read the warnings.

7.3 Assessing Credibility Online

Systems in the past have looked at other domains outside of software installation. For example, Schwarz and Morris have explored augmenting web pages and search results with visualizations presenting credibility features [16]. However, the credibility features, which were selected by the authors, are highly subjective and can often times provide security flaws. For example, the "popularity by experts" feature, which the study subjects found to be most useful, is questionable in terms of how the system selects the experts.

Fogg et al. have evaluated the credibility of two websites covering similar topics by applying the Prominence-Interpretation theory [12]. They found that the design of the site was the most frequently utilized aspect for evaluating credibility, followed by information structure and information focus. They also found that content is a factor that affects prominence as people notice different aspects when they examine different types of sites.

8. CONCLUSION

Users are often confronted with vexing trust decisions of whether to download a given piece of software. Unfortunately, automated techniques have proven inadequate so far, as malware continues to thrive. One solution is to provide users with additional *trust evidence* to enable them to make better trust decisions. IE's SmartScreen Filter (SSF) is a promising step in this direction. As we discover in the two user studies discussed in this paper, experts generally agree what additional trust evidence is required for making trust decisions, and novice users can make better trust decisions with such information, even improving on SSF with statistical significance. We hope that these results encourage further research in this important area to ultimately enable users to make correct trust decisions with high confidence.

9. ACKNOWLEDGMENTS

We gratefully thank Sara Kiesler for her insightful feedback and help with statistical analyses, and anonymous reviewers for their valuable comments.

This research was supported by CyLab at Carnegie Mellon under grants DAAD19-02-1-0389, and W911NF-09-1-0273 from the Army Research Office, by support from NSF under awards CCF-0424422, CNS-1040801, and IGERT Dge-0903659, and by Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Of-

fice. The views and conclusions contained here are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of ARO, CMU, NSF, or the U.S. Government or any of its agencies.

10. REFERENCES

[1] Microsoft Security Intelligence Report. http://download.microsoft.com/download/0/3/3/0331766E-3FC4-44E5-B1CA-2BDEB58211B8/Microsoft_Security_Intelligence_Report_volume_11_English.pdf, 2011.

[2] Sophos Security Threat Report 2011. http://www.sophos.com/sophos/docs/eng/papers/sophos-security-threat-report-2011-wpna.pdf, 2011.

[3] Sophos Security Threat Report 2012. http://www.sophos.com/medialibrary/PDFs/other/SophosSecurityThreatReport2012.pdf, 2012.

[4] This Is Watson. *IBM Journals of Research and Development*, May/Jul 2012.

[5] P. Ayyavu and C. Jensen. Integrating User Feedback with Heuristic Security and Privacy Management Systems. In *Proceedings of Proceedings of the annual SIGCHI conference on Human factors in computing systems*, 2011.

[6] C. Bravo-Lillo, L. F. Cranor, J. S. Downs, and S. Komanduri. Bridging the Gap in Computer Security Warnings. *IEEE Security and Privacy*, 2011.

[7] J. C. Brustoloni and R. Villamarin-Salomon. Improving Security Decisions with Polymorphic and Audited Dialogs. In *Proceedings of Symposium on Usable Privacy and Security (SOUPS)*, 2007.

[8] R. Dhamija, J. Tygar, and M. Hearst. Why Phishing Works. In *Proceedings of the annual SIGCHI conference on Human factors in computing systems*, 2006.

[9] J. S. Downs, M. B. Holbrook, and L. F. Cranor. Decision Strategies and Susceptibility to Phishing. In *Proceedings of Symposium on Usable Privacy and Security (SOUPS)*, 2006.

[10] S. Egelman. *Trust Me: Design Patterns for Constructing Trustworthy Trust Indicators*. PhD thesis, Carnegie Mellon University, 2009.

[11] S. Egelman, L. F. Cranor, and J. Hong. You've been warned: an empirical study of the effectiveness of web browser phishing warnings. In *Proceedings of the 26th annual SIGCHI conference on Human factors in computing systems*, 2008.

[12] B. Fogg, C. Soohoo, D. R. Danielson, L. Marable, J. Stanford, and E. R. Tauber. How Do Users Evaluate the Credibility of Web Sites? A Study with Over 2,500 Participants. In *Proceedings of the Conference on Designing for User Experiences (DUX)*, 2003.

[13] C. Kuo. *Reduction of End User Errors in the Design of Scalable, Secure Communication*. PhD thesis, Carnegie Mellon University, 2008.

[14] S. Motiee. Towards Supporting Users in Assessing the Risk in Privilege Elevation. Master's thesis, The University of British Columbia, 2011.

[15] P. O'Kane, S. Sezer, and K. McLaughlin. Obfuscation: The Hidden Malware. *IEEE Security & Privacy Magazine*, Sept. 2011.

[16] J. Schwarz and M. R. Morris. Augmenting web pages and search results to help people find trustworthy information online. In *Proceedings of the annual SIGCHI conference on Human factors in computing systems*, 2011.

[17] J. Sunshine, S. Egelman, H. Almuhimedi, N. Atri, and L. F. Cranor. Crying wolf: an empirical study of ssl warning effectiveness. In *Proceedings of the 18th conference on USENIX security symposium*, 2009.

[18] A. Vishwanath, T. Herath, R. Chen, J. Wang, and H. R. Rao. Why do people get phished? Testing individual differences in phishing vulnerability within an integrated, information processing model. *Decision Support Systems*, 2011.

[19] R. Wash. Folk Models of Home Computer Security. In *Proceedings of the Symposium on Usable Privacy and Security (SOUPS)*, 2010.

[20] M. S. Wogalter. *Handbook of Warnings*, chapter Communication-Human Information Processing (C-HIP) Model, pages 51–61. Lawrence Erlbaum Associates, 2006.