USENIX

THE ADVANCED COMPUTING
SYSTEMS ASSOCIATION

"I'm regretting that I hit run": In-situ Assessment of Potential Malware

Brandon Lit, Edward Crowder, and Hassan Khan, *University of Guelph;*
Daniel Vogel, *University of Waterloo*

https://www.usenix.org/conference/usenixsecurity25/presentation/lit

This paper is included in the Proceedings of the
34th USENIX Security Symposium.

August 13–15, 2025 • Seattle, WA, USA

978-1-939133-52-6

Open access to the Proceedings of the
34th USENIX Security Symposium is sponsored by USENIX.

"I'm regretting that I hit run": In-situ Assessment of Potential Malware

Brandon Lit
University of Guelph

Edward Crowder
University of Guelph

Hassan Khan
University of Guelph

Daniel Vogel
University of Waterloo

Abstract

We conduct the first ever two-session controlled lab study (n = 36) where end-users are prompted to install real benign and malicious software on a standard Windows laptop. The first session observes typical decision making strategies when participants assess software for potential threats without any instructions. The second session repeats the experiment after introducing an "enhanced task manager" application with information like CPU usage, files accessed, and network destination country to examine if decision making strategies change with more system-level information. The time, confidence, and accuracy to classify software as benign or malicious is recorded, along with participant comments using a "think-aloud" protocol. These comments form a dataset of 2,651 excerpts that are coded into four top-level categories of "indicators" with 25 sub-categories. These indicators provide a perspective into how end-users examine and analyze software in-situ. Overall, end-users are surprisingly accurate at classifying malware and become even better when provided with more process-level statistics. Our analysis uncovers common misconceptions, shows reliance on indicators that bad actors could circumvent, and provides actionable insights for software and operating system providers to improve user interfaces and notifications.

1 Introduction

In 2022 there were over a hundred million new malware applications reported [3]. Malware authors increasingly rely on deception and abuse of trust to remain undetected. For the first half of 2022, VirusTotal reported an increase in malware distribution through legitimate domains (2.5 million samples downloaded through Alexa Top 1,000), malware signed with valid digital certificates (over a million samples), and malware disguised as legitimate software [23]. With the increased volume and deceptive practices, technical solutions alone are insufficient to detect malware [8, 13]. Understanding how humans ascertain the nature of potentially malicious software is imperative as these deceptive techniques can mislead users who are following sound security advice.

While significant research has focused on technical solutions to detect malware, no work has explored strategies humans employ to decide if suspicious software is an actual threat. In stark contrast, human factors for phishing threats have been examined in great detail, resulting in the improvement of warnings, indicators, and training [20, 26]. Most modern operating systems have controls and tools for users to establish if software is trustworthy (e.g., Verified Publisher for Microsoft Windows), monitor process behaviour (e.g., Task Manager for Microsoft Windows), and contain potential damage (e.g., User Account Control for Microsoft Windows). However, the related work is scarce on whether users employ these controls or tools when establishing the legitimacy of software and if they do so effectively. Similarly, over-reliance on indicators or tools that can be easily manipulated by malware authors will require a fundamental rethinking of how the trustworthiness of software should be established.

Investigating how end-users understand and predict the nature of unknown software introduces several challenges. First, end-users have diverse backgrounds (e.g., accounting vs. IT) and even users with the same background may have varying knowledge of types of threats and experience with classification methods. Second, there are different types of malware threats with diverse behavioural signatures. This means classification requires knowledge of behavioural cues in the right context. Third, an ethically responsible study must ensure that users experience the behaviours of malware without infecting or damaging other users or machines. Finally, for ecologically valid findings, end users should experience both real threats and benign counterparts on bare metal machines for a meaningful baseline. The last two challenges in particular introduce significant logistical issues in study design.

We conducted what we believe is the first ever study to examine whether end-users can classify malware in-situ. 36 participants were recruited to form three experience groups: *basic* (users with little technology background), *intermediate* (users with technical background), and *advanced* (users

who are security analysts or researchers). Each participant completed two one-hour in-lab sessions. In each session, they used a Windows laptop to execute six software applications, classifying each as malicious or benign during the process. There were three real malware samples. In the first session, no external analysis tools were provided; the participants relied on what was available on the internet or from applications installed by default. In the second session, we provided an enhanced version of Task Manager with details about network connections (e.g., destination country and autonomous system) and specific files accessed by a process. We measured the overall classification accuracy and self-reported confidence, the time to classification, and think-aloud comments on what factors contributed to each participant's decision.

Our results suggest people across all levels of experience are surprisingly effective at malware classification. In the first session, 88% of malware was classified with a median decision time of four minutes, but the classification accuracy for benign software dropped to 62% with a median decision time of five minutes. In the second session, classification accuracy for malware was 94% with a median decision time of three minutes. Benign software classification accuracy remained lower at 66%, with a median decision time of four minutes.

We also examine what kinds of indicators participants used to make their classification. This is based on a qualitative analysis of 2,651 excerpts taken from participant think-aloud comments. Each excerpt was coded into four primary categories of indicators and 25 sub-factors that contributed to a participant's process to decide if the software was benign or malicious. The analysis shows that participants often rely on trusted indicators like system notifications with publisher information, process resource usage, network connection destination, and online searches about the software and its publisher. However, basic participants also relied on misleading indicators such as file names or had misconceptions about the verified publisher of the software.

Our work makes four contributions:

- The results of a lab-based study showing end-users are surprisingly good at identifying malware, but also frequently identify benign software as a threat.
- A categorization of secondary indicators employed by different types of end-users when assessing potential malware threats.
- How these indicators could be used by operating system providers to improve messaging and provide more targeted information to users.
- How end-user misconceptions about software trustworthiness could be incorporated into cybersecurity training and awareness programs.

2 Related Work

Several previous efforts have focused on understanding end-user mental models or folk models (mental models that are not necessarily accurate in the real world [25]) for security [9, 25, 27]. Similarly, mental models for security have been studied for specific demographics (including older adults and US citizens) [17, 28]. The mental models have been explored for viruses, hackers, data loss, and data exposure. The common themes identified include viruses being bad, viruses causing mischief and supporting crime, and viruses being buggy software. These models are useful for security researchers when designing solutions or offering recommendations but do not investigate in-situ malware classification. Researchers have also investigated specific threats through user studies and interviews, including phishing [20, 26], attacks on passwords [5, 6, 10], attacks on smart homes and IoT devices [33, 34], and attacks on Augmented Reality [11]. To this end, these efforts have greatly improved the state of security for end-users but are only tangentially related to our work as these investigations do not involve malware.

Most research directly related to malware classification has focused on how malware analysts analyze malware. Reverse engineering and decompilation of unknown binaries are critical aspects of a malware analyst's job. Through interviews and in-situ tasks, researchers have explored the strategies that malware analysts employ when reverse engineering malware [7, 15, 24] and evaluated the usability of existing [16] or improved [31] reverse engineering tools for malware analysis. Yong et al. [32] interviewed 21 professional malware analysts when reverse engineering malware and found five common workflows with associated challenges during different stages. Malware analysis is different from our work as the goal of the analyst is to understand *how* malware carries out its actions. On the other hand, we aim to understand how different users classify malware and benign applications.

For end-users, Levesque et al. [13] conducted a 4-month field study with 50 home users who used instrumented laptops to collect possible malware attacks and gather user behaviour. Unlike our work, the primary purpose of their research was to develop effective methodology to evaluate antivirus products. Spero et al. [21] examined end-user mental models of malware and regular software using a questionnaire and two diagramming exercises with 40 participants. The researchers asked participants to draw on their understanding of how a word processor and malware work. They found participants did not have a deep understanding of how malware functions because they regarded malware as fundamentally different than benign software. Unlike our work, they did not require users to classify unknown software in the context of using a real computer.

Most related work to our work is Aonzo et al.'s study using a malware classification game [2]. In the game, players classified unknown software using only offline sandbox reports of malware execution. This forced participants to rely on the fixed feature set (i.e., behavioural indicators) contained in the reports. They recruited 110 people (72 novices and 38 experts) to play the game and compete to achieve the highest classification accuracy. They found experts and novices base

their decisions on approximately the same features, including "network", "file system", and "signature". Our work also compares basic and advanced users for binary classification, but rather than rely on sandbox reports in a game setting, our participants actually examine and install real malware or benign software. Additionally, we identify the correct and incorrect usage of 4 primary and 25 secondary indicators used by basic, intermediate, and advanced participants. By enabling users to run real software on bare-metal machines (no virtual machines), we understand the indicators that stem from program execution (e.g., aesthetics) that basic users consider and also capture their time to classify malware.

3 Study Design

The goal of this study is to answer the research question: *"What strategies and indicators do end-users rely on to establish software legitimacy?"* We use a realistic task protocol that asks participants to install real binaries on a standard laptop. Six binaries are "sent" in sequence using a simulated chat interface. This task captures factors that users consider when inspecting a binary file, and then possibly executing it to install something or to run an application. We hypothesized that user performance depends on knowledge of lower-level system information. So, we repeat the protocol in a second session, but this time we also show the participant a tool to monitor system activity.

There are multiple challenges for the study design: (i) the choice of malware and benign samples; (ii) how the software samples are delivered to the participant; (iii) what kinds of system information could help users; and (iv) users have diverse technical and security backgrounds. Below, we justify design decisions and then outline the experiment protocol.

Choice of software. Each of the two study sessions used three malware and three benign software. We chose Microsoft Windows 10 as it is the most widely used operating system in 2023 [22]. For benign software, our goal was to find obscure software since choosing commonly used applications could introduce a confound from varying participant experience. We chose three categories of benign applications: printer driver, file sharing, and disk utility. We used two printer drivers from Brothersoft Industries with file names *install.exe* and *jd662w632aus.exe*. One of the printer drivers also requested permission to add a rule to the firewall. For file sharing applications, we chose SHAREit-KCWEB.exe and *FrostWire.exe* (P2P file sharing). We chose two disk utilities, *DiskView.exe* (provides a graphical map of the disk) and *CCleaner.exe* (disk cleaning utility). All applications except *FrostWire.exe* were signed by the providers. 9/36 participants reported being familiar with only a disk utility—*CCleaner.exe*. Appendix 4 provides the software version and publisher details.

For malware, we chose the LockBit Black Ransomware [19], Async RAT program [18] and XMRIG CoinMiner [30]. These malware were chosen as these are among the top threats in 2023 [4]. We compiled safe versions of the RAT and Cryptominer (instead of in-the-wild malware samples) to ensure that our machines were not participating in any nefarious activity. This required setting up our own C2 server. None of the malware binaries were signed. Note that while we used a variant of the same malware between the two sessions (see details below), participants were never informed whether they correctly classified a software or not. We discuss the possible learning effect in the limitations section. The details of the chosen malware are as follows:

- *Ransomware:* We built Lockbit Black Ransomware from its leaked builder [19]. This Ransomware had no user interface, and we did not add a UI wrapper as it is atypical of Ransomware. Once executed, the Ransomware would encrypt all files on the computer, change the desktop background, and change all encrypted file icons and extensions. Between the two sessions, we modified the versions such that different backgrounds, icons, and extensions appeared between different versions. The Ransomware did not have a UI and only showed two pop-ups—one for an unknown publisher and the other for a User Account Control (UAC) privilege escalation pop-up. These files were named *DiskCleaner.exe*, *Background Cleaner.exe* for the two sessions.
- *Remote Access Trojan (RAT).* The source for Async RAT [18] was recompiled, and the files were named *lsass.exe* and *services.exe*. This naming was adopted to simulate masquerading as known processes. For one session, the Command and control server (C2) was setup in Hong Kong, and for the other session, it was setup in the Russian Federation. Executing the RAT would only show a security warning about an unknown publisher.
- *Cryptominer:* The source for XMRIG CoinMiner [30] was compiled and embedded into two custom VB.Net loaders, designed to replicate common Trojan loader behaviour such as malware delivery via embedded payloads, dropping and executing payloads to temporary folders and spawning both child and separate processes from the Trojanized parent. The loaders were guised as a printer driver and the popular Adobe Creative Cloud Updater tool. The driver software presented an EULA (copied from the real driver) and a button to accept, after which a faux loading bar would appear before closing with the prompt that the installation was successful. The Cryptominer would launch while the loading bar was shown on screen. Adobe Creative Cloud Updater presented an Adobe loading animation alongside a loading bar, claiming to be downloading updates. The loading bar showed a Marquee continuous loading bar, while the Cryptominer launched in the background. For the two sessions, the files were named *Brothersoft_Driver223_Install.exe* and *creativecloud_cc_64_en_hi_gocd_mdr_install.exe*.

We carefully chose our benign samples to ensure they had behavioural similarities with malware. This makes the classification task challenging and enables us to study user strategies

better. For instance, Ransomware ramps up Disk Read/Write, so we use disk utilities. Similarly, RAT creates outbound connections, and our choice of file sharing software exhibited the same behaviour. Finally, the Cryptominer was Trojanized as a printer driver and Adobe Cloud updater. To demonstrate the overlap in behaviour, we use Zenbox Sandbox from VirusTotal to collect and compare MITRE ATT&CK Tactics and Techniques associated with our software samples. The analysis shows several overlapping T-s between the benign and malicious samples. The overlap is shown in Figure 9 (see Appendix A.5). While we did not Trojanize Ransomware and RAT (as the original samples did not come with a UI), we made realistic UI wrappers for Cryptominer to make it more challenging for the users. Finally, it should be noted that our choice of benign software is quite diverse, including software with firewall warnings and unverified publishers.

Delivery of software. The source of the software contributes to its trustworthiness (e.g., software downloaded from the website of the OS provider is generally trusted) [23]. We did not want the source of software to influence participants' classification decisions. Therefore, we developed a faux Microsoft Teams interface as a React Native application, which precisely mimicked the real Microsoft Teams interface and showed the files were from coworkers. A faux interface was used because the real Microsoft Teams interface would warn about the known malware.

Enhanced task manager. Examining characteristics of process behaviour could help end-users identify malicious or benign software. System process inspection tools, like Windows Task Manager and the advanced Process Hacker (2.4 million downloads in 2023 [29]), provide detailed information. However, they are complex for basic users to use, and understanding threat-related characteristics, like the origin of network connections and types of open files, requires additional information and knowledge external to the tool. Furthermore, it would have introduced a previous experience confound with our expert group, who are security professionals. We created a simpler and more targeted tool to inspect system process information that relates to threat characteristics (available as open source[1], screenshot in Appendix A.3). Adopting the tabular layout of Windows Task Manager, it lists process names with CPU%, memory usage, MB read and written to disk, and time since process started. It extends this information with destination countries associated with network connections (including autonomous systems that register an IP address), the verified publisher if one exists, and organizing files accessed by their parent directory. The information provided in the tool was informed by Aonzo et al. [2], who report "network", "processes", and "file system" are the top indicators used by advanced users in offline sandbox reports.

Participant background. Previous experience and prior knowledge could influence how end-users classify potential malware, so recruited participants were sorted into three groups. The *basic* group are end-users with no technical or security background and no previous post-secondary computer science education or training. The *intermediate* group are end-users with a computer science background or training but no security background. The *advanced* group are end-users with over two years of cybersecurity experience or an advanced degree in cybersecurity, typically working as security engineers or security analysts. The categorization was based on self-reported responses to education and experience related to computers and security. All participants reported they used Microsoft Windows at home or work for at least two hours a day on average, so they were familiar with operating system used in our study.

[1]Enhanced task manager: https://github.com/Brandon1234/enhanced-taskmanager

4 Methodology

Our methodology was shaped by a pilot study with 6 participants (4 basic, 1 intermediate, 1 advanced). Below, we discuss our methodology noting changes that resulted from the pilot.

Ethical considerations. Our institution's research ethics board examined our recruiting procedure, experimental protocol, the measures adopted for the security of users from interaction with malware, the measures to ensure participant anonymity and data confidentiality. All participants provided informed consent. They were told not to enter personal or sensitive information during the logged sessions, but just in case, they could withdraw their data within two weeks after study completion. Participants were informed that they should not use the device to enter any personal or sensitive information. The malware samples were recompiled to contact a "malicious server" that was under our control on a separate network, thereby ensuring that the malware will not spread to other machines. All participants took less than two hours to complete both study sessions on the same day and they were remunerated $60.

Participants. We advertised our study on Facebook marketplace, [anonymized regional online marketplace], LinkedIn, and through word-of-mouth. Using the participant background criteria described above, we sorted and screened respondents to form three groups of 12 people each. All 36 participants completed both study sessions. Each participant was remunerated $30 for each session. Figure 1 provides participant demographics, showing diversity in terms of age, daily computer use, organization size, and years of experience in IT or security. Participants were asked to describe their job title. The basic group reported: customer representative (3), administrative assistant (3), actor, entomologist, social worker, nurse, and other (2). The intermediate group reported: software developer (8), manager (2), IT, and analyst. The advanced group reported: software developer (3), threat analyst (7), architect, and IT.

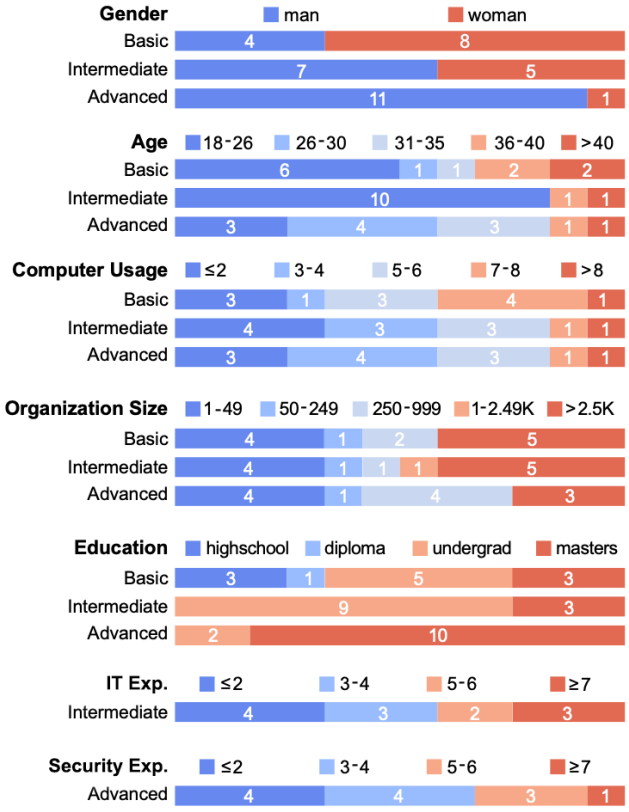


Figure 1: Participant demographics by group: computer usage is hours-per-day; Age, IT and security experience is years.

Apparatus. Our study required six malware, six benign applications, the enhanced task manager, and the Faux Microsoft Teams interface. Since our study required the execution of real malware in a controlled lab environment, we acquired four Windows 10 laptops (Lenovo Core i5, 8GB RAM, and 256 GB SSD). Multiple identical laptops enabled us to swap devices during the experiment to "reset" the system to a default state (we did not run virtual machines), which was especially important when the previous task resulted in the machine becoming infected with real malware. The laptops were configured identically using the default Windows installation, but we disabled Windows Defender Antivirus to avoid any warnings from it. Another configuration change was the result of the pilot study, where we noted that our UI wrappers for malware required additional files to be installed alongside the malware in order to execute. The UI wrapper files were made hidden to not confuse the participants.

Procedure. At the beginning of the first session, the participant completed a short demographic survey (see Appendix A.2.1). Then they were given the opportunity to familiarize themselves with the laptop. Throughout the study, a researcher sat nearby such that the laptop screen was visible. Microsoft Process Step Recorder logged low-level participant interactions, and the researcher noted the sequence of steps they took, including queries used in online searches.

Each participant was asked to imagine working at an enterprise company where the "faux Microsoft Teams" interface was used for employee communication. The interface presented unread messages from fictitious colleagues with unisex names. The pilot showed that specific message text could influence the participant's classification decision. Therefore, the message text was blurred and the participant was told to imagine a personal message from a coworker. Furthermore, we explained it was normal for coworkers to send files in this manner, but coworkers were not technologically competent.

For both sessions, the participant was not told how many samples were benign or malicious; they were only told they would be "sent" six different pieces of software. Both sessions were run together with a ten-minute break in between for participants. Running the two sessions consecutively also helped to reduce the learning effect on participants between sessions.

Procedure. For each software, the participant was asked to determine whether it was benign or malicious with the verbal instruction: "You are at work and you have just received a teams message from [the name of the coworker for that sample], they have told you that it is a [the assigned purpose of the software], your job is to determine whether or not you believe it to be legitimate or malicious software." The presentation order of the 6 software examples were counterbalanced across all participants using a balanced Latin square.

The participant was asked to concurrently *think-aloud* [1]: "speak out loud, or walk me through your thought process, anything you notice, and anything that you feel to be important as you work through the task". The participant was reminded to think-aloud if they did not speak for 10–20s with the prompt: "Remember to speak out your thought process."

After the task ended with a classification decision, the researcher reset the laptop to the default state or switched the laptop to a "clean" one. Then, the participant was asked to read the next simulated Microsoft Team message, which initiated the next software assessment task.

Enhanced task manager condition. The sessions were exactly the same, except in the second session, we introduced an enhanced version of the Windows Task Manager. At the beginning of the second session, participants were shown a slide deck describing the user interface of the enhanced task manager. This description did not include any suggestions or strategies for using the information to investigate potential malware. The researcher opened the enhanced task manager on the laptop before handing it back to the participants. Participants were encouraged to use the tool when classifying software. The instruction was: "As you can see the task manager software is already running on the device. You can take a moment to familiarize yourself with it and make any adjustments that you would like. Once you are ready I will give you your next task" (The possible adjustments include sorting options for different columns).

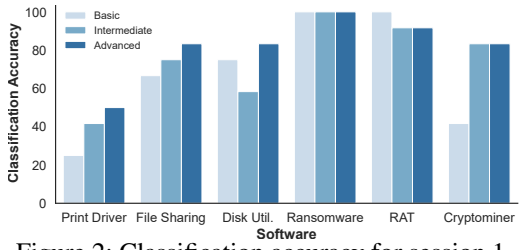


Figure 2: Classification accuracy for session 1.

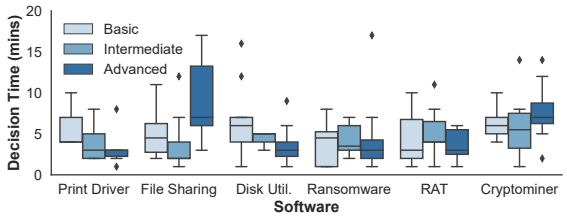


Figure 3: Decision time for correct classification in session 1.

We did not counterbalance sessions, meaning participants were always introduced to the enhanced task manager in the second session. This was done to avoid an asymmetric learning effect since participants will likely change their behaviour after being introduced to the enhanced task manager.

Analysis. The primary dependent variables are:
Classification Accuracy, defined as the ratio of true positives or true negatives over the number of malware or benign software samples, respectively (as %); and
Decision Time, is the time from when the participant opened the simulated Teams message until they provided their decision (wall clock time rounded to the next minute, which provides an upper bound on *Decision Time*).

We suggested to the participants that they spend less than five minutes examining each software. Most pilot participants completed the task within five minutes, and this suggestion kept the session duration reasonable. Note this was only a suggestion, participants were allowed to examine a software up to 20 minutes if they chose to do so.

We also performed qualitative analysis of the think-aloud comments. These were manually divided into 2,651 excerpts, each containing a single factor that the participant considered in their decision process.

We used an *inductive approach*, which is a "bottom-up" method to generate new codes from patterns in the data, rather than "top-down" where codes emerge from an existing theory [14]. The researchers met weekly during the study to discuss the emerging set of codes, halting recruitment when no new codes were emerging (i.e. no new codes emerged from the last two participants in each category). This resulted in a two-level codebook (see Table 1). The four primary codes relate to types of "indicators" that the participant used for their decision: *"program behaviour"*, *"program look and feel"*, *"executable properties"*, and *"threat intelligence sources"*. These are further divided into 25 second level codes. For each excerpt, we track the correctness of their rationale in the context of the software task (i.e., do they understand what the indicator represents in that specific context), see Appendix A.7). An example of an incorrect rationale by an advanced participant (P26): *"It was built for Windows 8 not Windows 10, which means it was made from the attacker's computer"*.

The coding process was as follows. First, 300 randomly chosen excerpts were assigned first-level codes by two researchers using formal coding rules (see Appendix A.6). This resulted in "substantial agreement"[2] between the two researchers (Cohen's $\kappa = 0.79$). The researchers met to discuss coding disagreements and found many were due to missing context in the excerpts, so these were updated accordingly. The two researchers performed another round of coding of 300 randomly chosen, non-overlapping excerpts, which resulted in "almost perfect agreement" (Cohen's $\kappa = 0.94$). The remaining excerpts were split between the two researchers who individually coded them. Second level coding was done after all first level codes were completed. Two researchers assigned second level codes to 300 random excerpts with "almost perfect agreement" (Cohen's $\kappa = 0.90$). The remaining excerpts were split between the two researchers and remaining second level codes assigned independently.

5 Results: Session 1

After reporting classification accuracy and time, we present qualitative results relating to contributing indicators.

Threat Classification Accuracy. Across all participants and all software, we found an overall classification accuracy of 75% (162/216) (Figure 2). For basic, intermediate, and expert participants, the overall classification accuracy is 68% (49/72), 75% (54/72), and 81% (61/72), respectively (Appendix A.1 provides more descriptive statistics for participant groups). The classification accuracy is 62% for benign software (67/108) and 88% for malware (95/108).

A Shapiro-Wilk normality test found classification accuracy deviated from normality (benign $p < 0.0001$, malware $p < .0001$), so a non-parametric Mann-Whitney U test was used. It indicates that classification accuracy for malware is significantly higher than benign software ($U = 7344$, $p < .0001$).

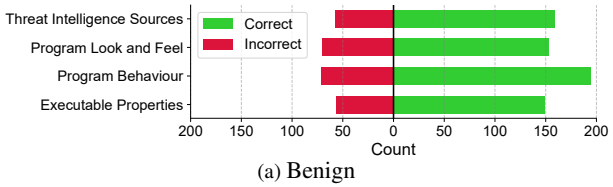
For different software, we note the same or comparable classification rates for the Ransomware and RAT between different participant groups. The advanced participants performed marginally better than basic (and intermediate users) for the File Sharing software (83% vs. 67%). They performed much better than basic participants for the Cryptominer (83% vs. 41%) and the Printer Driver (50% vs. 25%). Advanced participants outperformed the intermediate participants with the Disk Utility (83% vs. 58%).

[2]We provide suggested interpretations of Cohen's κ values [12]

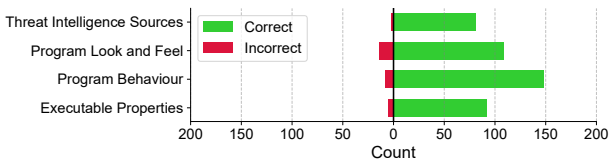
Table 1: Codebook (created inductively from participants comments).

Code	*Description*
Executable Properties	Information provided by the installation executable (not the running process)
Signature	Information about the signature certificate
Executable Metadata	Metadata provided by the executable
File Size	The file size of the executable
Publisher Reputation	Participant comments regarding who the executable claims to be signed or distributed by
Program Behaviour	Application behaviour after execution
System Notifications	OS dialog pop-ups, security notifications, and UAC prompts
Resource Usage	Usage of hardware resources (CPU, Memory, etc.)
UI Presence	Does a UI appear after execution?
File Access	The creation, deletion or editing of files by the application
Installation Error	Errors (only one application gave Java errors)
Customizability	Customization options in the installation setup
Execution Latency	Delays identified by participants during application execution
Informativeness	Providing status of installation
Program Look and Feel	All visual aspects before, during, and after running the application
Aesthetics	The visual appearance of the application
File Name	The name of the application
Typos and Grammar	Any typos or grammatical errors in the application
Icon	The icon of the application
Web Links	Any external weblinks that are visible during installation or execution
Eula/legal document	Did EULA/legal document appear during the installation
File Extension	The executable extension
Threat Intel. Sources	All third party sources of information
Online Search	Information found online by participants
Prior Knowledge	Information gathered by participants prior to and outside of the study
Network Connections	Network connections and network traffic
Online Scanning	Online scanning tools (e.g. VirusTotal)
Hash Lookup	Hash lookup using online sources
DNS	DNS lookup

Threat Classification Decision Time. Recall that we suggested participants spend up to five minutes on each sample but we did not enforce it. We present statistics for decision times for the readers to correlate the classification accuracy performance of participants with decision time. While many participants spent more than five minutes (see Figure 3), our suggestion may have influenced the decision time of participants. The median decision time for correct classifications across all software is 4 mins (M 5.1; SD 3.7), with times for specific types of software shown in Figure 3. In comparison, the median decision time for incorrect classification for all software is 5.5 mins (M 6.3; SD 3.9). Considering benign and malicious software separately, the decision time was 5 mins (M 5.8; SD 4) and 4 mins (M 5.1; SD 3.5). Between the basic, intermediate, and advanced participants, for correct decision across all software, we note a median decision time of 4.5 mins (M 5.1; SD 3.6), 4.5 mins (M 45.4; SD 3.4), and 4 mins (M 5.3; SD 4.1), respectively.



(a) Benign



(b) Malware

Figure 4: First level indicators reported by participants for session 1. Correct/incorrect indicate whether participants applied the indicator correctly in the given context.

5.1 Contributing Indicators

Figure 4 summarizes the counts of the first level indicators for benign and malware software, including whether the associated classification decision was incorrect or correct. Participants reported 1,373 indicators across all 216 software classification tasks. The median number of indicators for each participant was 38 (M 39; SD 15). Participants commented on using 912 indicators for benign software: 72% (656/912) correctly and 28% (256/912) incorrectly. Participants correctly used *"executable properties"* as an indicator 149 times, *"program behaviour"* 195 times, *"program look and feel"* 152 times, and *"threat intelligence"* 159 times. Participants incorrectly used *"executable properties"* 57 times, *"program behaviour"* 71 times, *"program look and feel"* 70 times, and *"threat intelligence sources"* 58 times. 461 indicators were reported for malware: 94% (432/461) correct, 6% (29/461) incorrect. Across the four top-level indicators, participants correctly used *"executable properties"* 92 times, *"program behaviour"* 148 times, *"program look and feel"* 109 times, and *"threat intelligence sources"* 81 times. These indicators were only used incorrectly 14 times or fewer for the top-level indicator categories for specific malware software examples.

We now discuss how each participant employed second level indicators in their decision making, organized by first level codes. Due to space constraints, we only report the number of participants that correctly or incorrectly employed an indicator in Table 2 and do not report the counts in the text (see correct and incorrect code usage examples in Appendix A.7). When reporting quotes from participants to represent a theme, we identify the number of participants who expressed that code and provide a representative quote. We identify participants with their group (**B**asic, **I**ntermediate, or **A**dvanced) and the unique identifier assigned to them.

Table 2: First and second level of reported indicators (Correct/Total) by participants for both sessions. (Inter. = Intermediate; Adv. = Advanced)

	Malware			Benign		
	Basic	**Inter.**	**Adv.**	**Basic**	**Inter.**	**Adv.**
Executable Properties						
Signature	34/39	27/29	26/28	25/35	21/43	25/35
Executable Metadata	19/19	36/36	30/33	12/22	24/39	27/43
File Size	8/13	24/25	22/25	10/16	11/22	14/20
Publisher Reputation	7/7	12/13	6/6	5/10	3/9	5/6
Program Behaviour						
System Notifications	31/34	42/42	26/27	21/42	21/44	23/38
Resource Usage	24/31	59/61	37/39	8/14	22/29	15/22
UI Presence	23/24	20/20	19/19	5/9	3/5	2/3
File Access	31/31	23/23	20/20	6/6	2/7	1/2
Installation Error	7/10	9/9	6/6	23/38	12/25	12/15
Customizability	0/1	2/3	1/1	6/18	10/18	6/11
Execution Latency	10/15	13/14	1/2	8/13	3/3	1/1
Informativeness	3/5	3/3	2/2	7/12	5/8	2/4
Program Look and Feel						
Aesthetics	12/20	16/17	16/20	26/36	29/40	19/30
File Name	35/45	42/44	25/26	28/44	22/42	20/31
Typos and Grammar	1/1	0/0	0/0	5/11	5/15	2/2
Icon	8/12	15/16	7/8	20/27	18/33	10/15
Web Links	2/2	2/2	0/0	6/7	6/8	0/1
EULA/Legal Doc.	9/12	14/15	3/5	8/15	4/7	2/3
File Extension	4/6	6/6	3/3	3/5	5/8	1/2
Threat Intelligence Sources						
Online Search	15/22	37/37	35/36	31/36	36/65	37/60
Prior Knowledge	19/23	32/32	22/24	13/19	19/44	23/48
Network Connections	13/14	24/24	19/19	3/4	3/9	4/10
Online Scanning	0/0	0/0	20/20	0/0	2/4	18/28
Hash Lookup	0/1	0/0	1/1	0/0	0/0	4/4
DNS	0/0	0/0	1/1	0/0	0/0	1/1

5.1.1 Executable Properties

Signature. Intermediate and advanced participants actively looked for the digital signature while most basic participants stumbled on it while checking the properties or on execution (when the operating system presented them with a dialog containing a publisher). Participants from the three groups' associated signatures to be an indicator for benign. Understandably, two basic participants did not understand signature's significance or whether the certificate was expired, for example: *"[after opening the certificate] not that I know how to verify it anyway"* (B07). Ten basic participants were wary of unknown publishers *"whenever something blocks it because it doesn't have a valid signature or no publisher, I get suspicious"* (B10) even if they did not have a deeper understanding of how they are verified *"I feel like Microsoft should know the publisher, still it could be a lesser known publisher like a non for profit but still"* (B12). Four intermediate participants did not consider that the digital signature could be from a compromised signature key. Their comments were similar to: *"there's no way to forge a Microsoft one but I don't know enough about certificates to know [if that is correct]"* (I16). Five advanced participants noted the susceptibility of signed applications. Their comments for the *CCleaner.exe* application were similar to: *"[after noticing the signature from Microsoft] doesn't mean much nowadays"* (A33).

Executable metadata. Participants inspected executable metadata by right clicking and selecting "Properties". Microsoft Windows also makes the metadata available on mouse over event and nine basic participants used this information. All but twelve participants from the three groups attributed the presence of metadata as an indicator for benign. Their comments were similar to: *"[there are] more details than it*

would [have] if it were spammy" (B11) and *"a virus would not have that information"* (I18). Nine participants viewed missing metadata to be negative indicator. Their comments were similar to: *"[there is] very little information here"* (A30) and *"I would expect it to at least have the name of the company saying who owns it, where it is from, [and] what it is rather than just a random executable"* (I21). Finally, we noted that seven participants (four advanced, two intermediate, and one basic participant) reported that the file version of 1.0 in the metadata seemed suspicious.

File size. 22 participants from all three groups looked at the file size. While basic and intermediate participants commented on the size being big or small, e.g. *"A lot of KB for installing something"* (B02), six could not comment on how it helped their decision: *"I don't know what it means [for it to be this large] but it is"* (I13). Advanced participants had different plausible reasons for what file size meant. Their comments were similar to: *"these days size doesn't help as much"* (A37), and *"[large file size meant] a malicious program could be bundled in there"* (A29). One basic, six intermediate, and three advanced participants compared the file size with an online source and if it matched, they considered it to be a sign for benign.

Publisher reputation. Based on their past experiences, participants exhibited positive or negative bias based on the publishers' reputation. Basic participants trusted the company if it was well-known, e.g. *"I trust Microsoft a lot"* (B04), and were suspicious if they had not heard of it, e.g. *"[The] publisher [Brothersoft] is weird" (B06).* Intermediate and advanced participants judged the software quality relative to the publishing company. The installation wizard for the printer drivers was outdated which was expected by six participants. Their comments were similar to: *"It's a printer company, we cannot expect anything [of quality] from them"* (A37).

5.1.2 Program Behaviour

Program behaviour entails several aspects that participants noted after executing the installer or the application itself. Note that we categorized *"network connections"* and *"DNS"* under *"threat intelligence sources"* since advanced participants consulted external sources to check their reputation.

Installation error. Participants encountered unexpected behaviour during software installation or launching application for the first time after installation. (Note that there were no actual installation errors except a Java version error for an application used in session 2). For instance, five basic participants were confused when the applications would not launch automatically after installation. Their comments were similar to: *"I don't know what's happening and I'm worried about it"* (B01). Six intermediate and advanced participants were able to comment on the sources of confusion: *"This is not very straight forward, and the structure of the folder are odd [as they analyze the installed folder for the Frostwire application]"* (A27). However, participants were unsure whether to attribute the unexpected behaviour to poorly developed benign software or malware.

Customizability. All participants who reported customization options (installation directories, add shortcuts, and cancel the installation) during the installation attributed customizability to benign software.

Informativeness. During the installation, expected UI elements (e.g., progress indicators or required disk space) were associated with benign behaviour. The only exception was a dialog displayed by the printer to add a firewall exception, which was concerning for nine participants. Their comments were similar to: *"Why does a printer need exceptions for your firewall?"* (I21).

UI presence. After the installation, some installers automatically launched the application with a UI while others did not. When any participant did not see a UI, they would become suspicious. The cases that were attributed to installation error are covered under *"Installation Error"* code.*"UI Presence"* is for the cases where participants reported that they did not see a UI because the software had no UI. Furthermore, the lack of UI was considered suspicious. Their comments were similar to: *"it is not inherently, visibly doing anything"* (B08), and *"If it's some sort of file sharing something I would expect there to be some sort of UI"* (I21).

System notification. Participants encountered different notification dialogs including UAC and Security Notification (informing about the verified publisher). Basic participants were suspicious when the UAC notification or unsigned software warnings appeared. Their comments were similar to: *"whenever something blocks it because [the software] doesn't have a valid signature or no publisher, I get suspicious"* (B10). Five basic and two intermediate participants also dismissed unverified publisher warnings with reasons similar to: *"[I've] seen unverified publisher warnings before so it could still be ok"* (B03) and *"[During software development, I've] been there done that"* (I15) while still being wary about them appearing. Advanced participants were expecting these notifications as they already investigated the publisher.

Execution latency. Some benign and malware samples took longer to execute. When faced with this delay, basic participants would tend to express confusion, assuming that they simply had to wait longer *"I guess it's still just downloading"* (B03). However, after waiting for a couple of minutes, six participants grew suspicious and were inclined to question the legitimacy of the software *"the longer it takes the more confident I am that this is malware"* (I14).

Resource usage. Participants noted resource usage during application execution. Intermediate and advanced participants used Windows Task Manager and noise from the system fan while basic participants only noticed the lat-

ter. Seven basic participants associated more resource usage with malware with comments similar to: "[The computer] is super slow" (B01). Nine intermediate and eight advanced participants paid attention to other indicators including CPU/memory/disk/network usage and running processes.

File access. Participants were provided with no tool that informed them about the file access for session 1. Therefore, all participants were only able to notice suspicious file modification by the ransomware after its execution. All participants recognized that it modified the files and it was malware.

5.1.3 Program Look and Feel

File name. Participants saw the filename from the Teams message. Basic participants construed a lot from the filenames that were not human readable. Comments were similar to: *"[Name lsass.exe is] like wearing a name tag that says robber to a bank"* (B09). Intermediate and advanced participants relied on their experiences. Descriptive names were an indicator for benign, and less descriptive names were an indicator for malware. Comments were similar to: *"[creative-cloud_cc_64_en_hi_gocd_mdr_install.exe (Cryptominer) is] a little too inclusive of details such as the platform and release information"* (I19) and *"no file in history has been called that ["install.exe"] and not been malicious"* (A02). However, both intermediate and advanced participants recognized filenames could be easily changed *"you probably shouldn't be taking too much information from the name"* (A29).

Aesthetics. Participants executed the software and noted its aesthetics ranging from the shape and size of the window to the styling of buttons and text boxes. Basic participants relied on their intuition and past experiences when judging aesthetics. However, they were not able to verbalize what seemed suspicious. Their comments were similar to: *"something is wrong I don't know what it is though"* (B01) or *"[it is] not what the Windows window should look like"* (B06). Intermediate and advanced participants were more specific with comments like: *"The UI looks weird, crude, and unpolished"* (I21). Participants also leveraged past experiences with similar categories of software. For instance, when inspecting a printer driver, the comments of six participants were similar to: *"this looks like every single other driver software I've seen so I'm pretty happy about it"* (A35).

File extension. All three participant groups had participants that noted the extension. However, only basic participants treated the extension as a significant factor to consider, intermediate and advanced participants expressed concern but used alternative indicators to justify their evaluation. Four basic participants used the file extension in this way. Their comments were similar to: *"you have to be really careful with those and if I don't know it's from a reliable source I would not want to open it"* (B12) and *"seeing the .exe made it more computer-y"* (B04). 10 participants from the other groups noted the dangers of executables and recognized the software nature of these files (unlike data files).

Icon. Basic participants relied on the icon more than the other groups. Furthermore, several basic participants mistook the default Windows executable icon as the icon from the publisher. A more alarming finding was that six participants (from basic and intermediate groups) were confused about the UAC shield overlaid on top of the software icon. Half were unsure what the shield meant and the remaining half attributed it to a secure software. Their comments were similar to: *"Shield on the installer [the executable] is a good sign"* (I15). On the other hand, most intermediate and advanced participants understood that the overlaid shield represented elevation of privileges.

Typos and grammar. After launching the application participants looked for grammar or typographic errors in the installation process and the GUI of the program. All participants associated the presence of typos with malware in the installation or software interface and their comments were similar to: *"Incomplete sentences are bad"* (I18).

EULA/Legal documents. As the software that had installers ran, participants would encounter an EULA agreement that they had to accept. Across all three groups participants either stated that they would not normally look at it, or that they would normally skim through it without too much detail. Of the participants who viewed the EULA agreements, they attempted to compare what they were reading with their past experiences *"This is looking a lot more like what I'm used to when installing printer drivers"* (I21).

Web links. Participants spotted social media links during the installation or within the application UI. Most basic participants reported that these links were a good indicator of software legitimacy. Unlike basic participants, intermediate and advanced participants opened the link to gather more evidence on the software/publishers' reputation. The publisher for "DiskView.exe" was Microsoft but it had a link to SysInternals website (acquired by Microsoft). This discrepancy was suspicious to five participants.

5.1.4 Threat Intelligence Sources

Online search. Participants used online search engines. Six basic participants only searched the name of the executable or the publisher and relied on the results to make the decision. Their comments were similar to: *"[After finding enough content on the publisher's Wikipedia page] There is enough stuff in the right places"* (B07) and *"[After seeing relevant links] it is something legitimate"* (B05). Nine intermediate participants also looked for legitimate download pages containing the software. Eleven advanced participants also expected meaningful search results and were doubtful about a software's legitimacy if they could not find any relevant

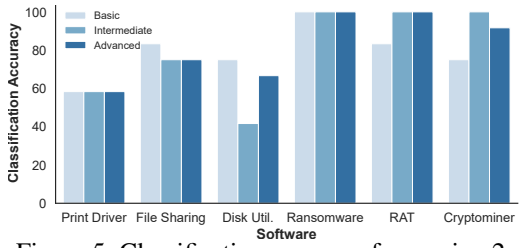


Figure 5: Classification accuracy for session 2.

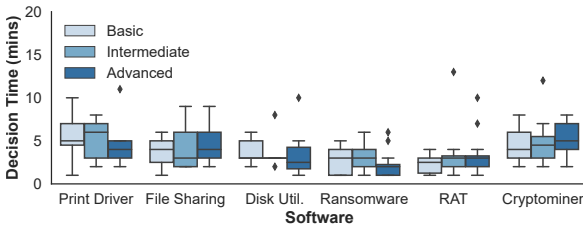


Figure 6: Decision time for correct classification in session 2.

information *"[After searching the executable name] There are no big companies taking claim of that executable"*(A31).

Prior knowledge. Participants relied on their prior knowledge and it influenced their decision. Almost all participants had negative connotations with disk utilities. Their comments were similar to: *"[After seeing the word disk in the name] it is a scam"* (I26). Intermediate participants relied on their development experience and noted different aspects surrounding it. For instance, one participant commented, *"It looks like a messy program location with a lot of command switches after"* (I19). They also had expectations for different software categories *"Printers take weird things to work"* (I17). Advanced participants had varying expectations based on their experiences. Examples include: *"Nowadays executables are not what you would want [for an installer]"* (A32) and *"Request for more privileges is bad"* (A30).

Online scanning. Seven advanced participants scanned the executables at *VirusTotal* or *Joe's Sandbox*. These participants considered their verdict to be a reliable indicator.

Hash lookup. The online tools also returned a hash. Three advanced participants searched the hash online to find another copy of the software. If the hash did not match a known software, they treated it as suspicious.

Network connections. Participants did not have a tool to see individual network connections for session 1. No basic participant looked for this indicator, but intermediate and advanced participants looked for spikes in network traffic. They also noted that they would want to install an external tool to understand the network connections.

DNS. Two advanced participants looked at the DNS cache of the device before and after executing the software to see if the software was connecting to any suspicious domains.

6 Results: Session 2

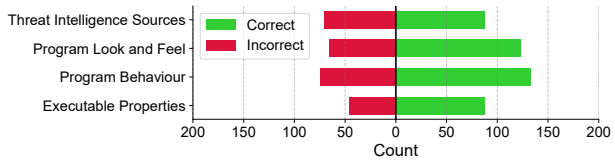
Threat Classification Accuracy. Figure 5 shows the classification accuracy of participants for the second session. We note the overall classification accuracy across all software is 80% (173/216). For basic, intermediate, and advanced participants, the overall classification accuracy is 79% (57/72), 79% (57/72), and 82% (59/72), respectively. For benign software, we note a classification accuracy of 65.7% (71/108) and for malware, it is 94% (102/108). (See Appendix A.1 for statistics on classification accuracy across participants.) A Shapiro-Wilk normality test found classification accuracy deviated from normality (benign $p < 0.0001$, malware $p < 0.0001$), so a non-parametric Mann-Whitney U test was performed. It indicates that the classification accuracy for malware is significantly higher than benign software ($U = 7506$, $p < 0.0001$).

For different software, we note the same or comparable classification accuracy for the Printer Driver, Ransomware, and File Sharing between different participant groups. The intermediate users perform better than basic (and advanced users) for the Cryptominer (100% vs. 75%), both intermediate and advanced users outperform basic users for RAT (100% vs. 83%), and basic users outperformed advanced and intermediate users when evaluating the Disk Utility (75% vs. 41%) classification accuracy.

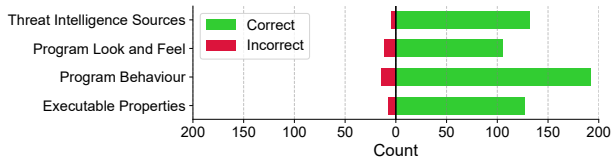
Threat Classification Decision Time. Figure 6 shows the decision time by the participants to classify software (for correct classifications only). It shows that the median decision time for correct classification for all software is 4 mins (M 4.4; SD 2.9). (In comparison the decision time for incorrect classification for all software is 4 mins (M 5.7; SD 4.2).) For benign and malicious software, we note a decision time of 4 mins (M 4.6; SD 2.4) and 3 mins (M 4.2; SD 3.2), respectively. For the three groups of participants across all software for correct classification, each has a decision time of 3 mins (M 4; SD 2.5), 3 mins (M 4.1; SD 2.5), and 4 mins (M 4.9; SD 3.6), respectively.

6.1 Contributing Indicators

Figure 7 shows the outcome of our coding exercise for session 2. For session 2, participants reported 1,280 indicators for the 216 investigations. The median indicators reported by each user for the second session were 35 (M 37; SD 12). Participants reported 688 indicators for benign software. 63% (432/688) were correct while 37% (256/688) were incorrect. Across the four indicators, *"executable properties"* was correctly identified 88 times, *"program behaviour"* was correctly identified 133 times, *"program look and feel"* was correctly identified 123 times, and *"threat intelligence sources"* was correctly identified 88 times. The reported indicators for benign were incorrect 256 times for malware samples. Across the indicators, participants incorrectly used *"executable properties"* 57



(a) Benign



(b) Malware

Figure 7: First level indicators reported by participants for session 1. Correct/incorrect indicate whether participants applied the indicator correctly/incorrectly in the given context.

times, *"program behaviour"* 71 times, *"program look and feel"* 70, and *"threat intelligence sources"* 58 times. 461 indicators were reported for malware—94% (432/461) were correct while 6% (29/461) were incorrect. Across the four top level indicators, participants correctly used *"executable properties"* 92 times, *"program behaviour"* 148 times, *"program look and feel"* 109 times, and *"threat intelligence sources"* 81 times. These codes were only incorrect 14 times or fewer for different categories of malware samples. We discuss the distribution across secondary codes and different types of software in § 7.

For the second level codes, we find qualitatively similar results for most second level codes except *"resource usage"*, *"file access"*, and *"network connections"*. With the enhanced task manager, all participants had visibility into these indicators. Therefore, we only report how participants employed these indicators for session 2.

Resource usage. For session 2, basic participants continued to the machine fans as an indicator and also obtained resource usage information from our tool. However, two participants were unsure how to interpret resource usage. Their comments were similar to: *"What does 21 CPU % mean?"* (B08). Intermediate and advanced participants were familiar with these metrics and simply used our tool to pull this information.

File access. 30 participants noted that the Ransomware was accessing a lot of files. We also noted that participants belonging to different groups provided plausible reasons for why a software needed to access local files. For the Cryptominer disguised as Adobe Cloud Updater, one basic participant commented: *"File access is not atypical as it is a cloud storage platform"* (B11).

Network connections. All three participant groups used the network destination country to question the behaviour of applications. Their comments were similar to: *"Why would I need to connect to Hong Kong?"* (B03), *"I don't like [seeing] a connection to a city [Hong Kong]"* (I15). Advanced participants demonstrated a greater understanding with opinions similar to: *"Why would it ping anywhere that is not Canada or US. This is probably a gateway into China, Russia or something like that."* (A36)

7 Discussion

We now discuss how different indicators were used to classify different types of software in session 1. We also discuss common misconceptions and areas that need more research.

7.1 Performance on Different Software

Basic. Participants from the basic group accurately classified both the Ransomware and RAT, but struggled to classify the Cryptominer. For the Ransomware, they became suspicious after observing *"system notification"* (14 times). When analyzing the RAT, participants reported lack of *"UI presence"* as the most common reason for their verdict (14 times). Basic participants were extremely sensitive to whether or not a UI would appear, though there was general confusion over whether they had made a mistake while attempting to run the application or if the application was broken or malicious. The high false positive rate for the Cryptominer is due to its Trojan nature. This is supported by the indicators—*"EULA/Legal notice"* (6 times), followed by *"aesthetics"* (5 times).

For benign software, we note a high false positive rate. The indicators that contributed to incorrect verdicts primarily focused on any received *"system notifications"* (reported 16 times), the *"executable metadata"* (reported 10 times), and the *"aesthetics"* (reported 9 times). For the correctly classified benign applications, the top contributing indicators include *"online search"* (19 times), the *"file name"* (15 times), and *"aesthetics"* (11 times). In both the correct and incorrect verdicts, the aesthetics of software played a significant role.

Intermediate. Participants from the intermediate group accurately classified the Ransomware, Cryptominer, and the RAT. For the Ransomware, they correctly detected the changes to files and the OS Desktop. The UAC notification also contributed to their lack of trust and resulted in the *"system notification"* indicator being reported 12 times, followed by *"prior knowledge"* (9 times). When evaluating the RAT, they used Windows Task Manager to determine the process's *"resource usage"* (15 times), and *"UI presence"* (14 times). When evaluating the Cryptominer, participants detected it using *"resource usage"* (reported 16 times). Only three participants from the intermediate group did not classify the Cryptominer. Their rationale was that during software installation, the CPU usage is high and no verified publisher is not a negative indicator. Their online search relied on the publisher or downloading the same software for comparison.

When investigating malware, intermediate participants' top indicators were *"resource usage"*, *"system notification"*, and the *"file name"*. They were alarmed by notifications with "red shield" or "yellow shield", and were able to tell when malware disguised as Windows or other known processes.

For benign software, similar to other groups, we observe a high false positive rate. The indicators that contributed to incorrect verdicts include *"system notification"* (18 times), *"online search"* (14 times), *"previous knowledge"* and *"signature"* (both 13 times). Participants tended to distrust the one benign software with an unknown publisher. The aesthetics of the website of the printer manufacturer and the information on the file sharing software contributed to the negative opinion about the software. For the correctly classified benign applications, the contributing indicators in session one include *"online search"* (24 times), and *"aesthetics"* (17 times), with *"resource usage"*, *"executable metadata"*, and *"file name"* all tied for third (13 times). These participants utilized online searches in the same way, but either augmented their searches or ignored irrelevant articles or issues with other versions of the software.

Advanced. Participants from the advanced group accurately classified the Ransomware and RAT, with only three participants failing to classify the Cryptominer. Advanced users were able to classify the Ransomware and RAT in the first session because they noticed *"file access"* (12 times), *'"system notification"*, and *"online search"* (6 times each) for the Ransomware, and *"resource usage"* (9 times) and *"UI presence"* (7 times) for the RAT. The indicators that contributed most to the incorrect decision for the Cryptominer in session one were the *"aesthetics"* (4 times), with the *"signature"*, *"file size"*, *"executable metadata"*, and the *"EULA/Legal document"* all reported twice.

For benign software in session one, the indicators that contributed to incorrect verdicts included *"prior knowledge"* (16 times), *"system notification"* (10 times), and *"online search"* (9 times). The majority of false positives were due to the confusion caused by their prior knowledge. Participants either had negative experiences with alternate versions or applications with similar names. Due to this history, they tried to find indicators that would stoke their suspicion (e.g., fixating on information that was absent in metadata or in a system notification). When they chose to search for information online, participants continued to carry their bias into their searches, affecting their final perception of the software. For the correctly classified benign applications, the contributing indicators included *"online search"* (22 times), *"executable metadata"* (18 times), and *"signature"* (14 times).

7.2 Misconceptions and Awareness Deficits

Signatures and security notifications. The findings in § 5.1 highlight several misconceptions among basic and intermediate participants. It showed that while some basic participants understood that security warnings were not a good sign, they were uncertain why they were seeing those warnings. After choosing to trust one of the malware without a publisher, one participant stated that *"I feel like Microsoft should know the publisher, still it could be a lesser-known publisher like a non-profit but still"* (B12). Other participants would note that there was no publisher but then continued, only associating the lack of a publisher as a negative thing explicitly after viewing the malware run. We note that the warnings were helpful as only four basic participants were able to recognize when a signature was not present in the executable properties. We note that six participants attributed the overlaid shield on the executable icon (for escalated privileges) to a secure software, which requires rethinking of the attribution of UI elements. Awareness programs should highlight the risks associated with installing unsigned applications.

File names. For basic participants, we noted that file names represent a more "truthful description" rather than something that could be modified to anything. When commenting on file names, they judged everything—from the presence of capital letters and underscores, to whether the name sounded "believable". While intermediate participants relied on file names too, they often compared their experience of creating official company applications to the file names. The presence of language and architecture ("en_64") in malware was considered an indicator of benign applications. Awareness programs should educate basic and intermediate users about the risks of ignoring security warnings and how to identify strong security indicators.

Program look and feel. § 5.1 discusses how participants noticed the aesthetics of the software, specifically looking to see how legitimate installers looked (e.g., check boxes in an installer). We also noted that basic participants were quite focused on spotting typos and grammatical issues. The EULA of a benign application had a typo that was caught by participants. On the other hand, the UI wrapper for the Cryptominer was quite polished, which deceived several participants: *"[It] looks like a legitimate Adobe installer"* (I18). The mental models associated with certain UI elements may also result in inaction from users. For instance, a basic participant kept waiting for the loading bar on the Cryptominer until the researcher asked them to move to the next software *"I guess it is still just downloading"* (B03). While prior experience may adjust expectations around the look and feel of the software (e.g., the reported low expectations for aesthetics from printer drivers), the recent deceptive practices from attackers show an increased exploitation of how programs are expected to look and UI elements indicate. Malware is increasingly being bundled with benign applications or disguised as popular benign applications [23]. While expecting the awareness programs to educate basic and intermediate users about the deceptive practices employed by the attackers may be too

ambitious, technical controls should be developed to detect masquerading attempts by unsigned software.

Indicator accessibility. The focus of our work is to study the malware classification capabilities of users, and not to design a tool that helps users classify malware. Nevertheless, we share observations from session 2 regarding the accessibility of the indicators. During the study session, most basic and intermediate participants reported that they found the enhanced task manager was helpful for their investigation. One intermediate participant stated that *"It [the enhanced task manager] was overall useful, specifically having the verified publisher, network connections and disk read/write. It's good to know, to see, if it [a process] was accessing large amounts of files or connecting to a location you wouldn't suspect."* (I18) The utility of the enhanced task manager with basic and intermediate participants suggests the need for making such enriched indicators readily available in the existing Windows Task Manager coupled with a better notification mechanisms to enable interested users to investigate potentially anomalous behavior by software.

8 Limitations

Like most studies involving human participants, our research has known inherent limitations that are widely acknowledged. Our survey relies on self-reported data, which are subject to participant memory, comprehension, and subjective perspectives. It is possible that participants may have inaccurately reported certain aspects, to avoid embarrassment or to present responses perceived as favorable to the researchers. Other limitations particular to our study are discussed below.

In terms of ecological validity, it is unclear whether users look for indicators when they download software from various sources. Since our goal was to understand the strategies of the user, we instructed them to classify malware. It is well accepted that end-users can be tricked into executing software on their device. Furthermore, the investigation that users may perform is also dependent on how the software was delivered. In our case, the messages in the Faux Microsoft Teams interface were blurred to ensure that users did not create a preconceived notion about the nature of the software based on the message. However, different delivery sources (online vs. P2P vs. colleague) may have different trust levels associated with them, which would change the investigation strategy (if any). Future work can explore the relationship between the source of software and the alertness of users to the reported indicators in their normal workflow.

Participants were informed that they had to identify malware. Furthermore, for our experiment, malware had a higher base rate than what is observed in the wild. Consequently, participants were primed to label samples as malware. Despite this limitation, our qualitative results provide valuable insights into user investigation.

Participants conducted the experiment on a laptop that they were not familiar with. Despite our instructions to get familiarized with the device, the unfamiliarity may have caused undesirable effects. However, this study would not be possible on participants' devices. Similarly, the advanced participants were severely constrained. Several advanced participants desired to have their own malware reversing tools to analyze the executable, which was not possible due to time constraints (note that § 2 discusses related works that explore strategies of malware analysts using their tools). Nine participants were familiar with one benign software while the rest were not. This may have resulted in better performance of the participants who were familiar with the software.

Variants of the same malware were used for the two study sessions. This reuse was unavoidable as the same class of malware had similar behaviour and using the same class of malware was important for a comparison. This may have resulted in a learning effect. To overcome this effect, we chose not to inform the participants about the correctness of their decision for either session. It should also be noted that despite our efforts, the benign software was not quite similar across the two sessions (e.g., both disk utilities were quite different than each other). While the results of individual sessions hold, these differences underscore the need for a cautious interpretation of the comparisons.

9 Conclusion

We conduct the first ever study to measure how humans approach malware classification. Our lab study employs real users against real malware to measure their classification accuracy and time. Through our analysis of over 2,500 comments, we uncover 25 indicators that humans use when assessing the legitimacy of a potentially malicious software. Our extensive analysis reveals that humans rely on a spectrum of security indicators, from more reliable tools like online scanners to less reliable cues such as file names. We uncover misconceptions about signed executables, UAC shield overlaying program icons, and the forgeability of verified publishers. Our findings are useful for security awareness programs and OS developers to eradicate misconceptions and improve security related interfaces and notifications.

9.1 Acknowledgements

This material is based upon work supported by NSERC under Grant No. RGPIN-2019-05120.

References

[1] O. Alhadreti and P. Mayhew, "Rethinking thinking aloud: A comparison of three think-aloud protocols,"

in *Proceedings of the 2018 CHI conference on human factors in computing systems*, 2018.

[2] S. Aonzo, Y. Han, A. Mantovani, and D. Balzarotti, "Humans vs. machines in malware classification," in *Proceedings of USENIX Security*, 2023.

[3] AV-Test Institute GmbH, "Malware Statistics," https://www.av-test.org/en/statistics/malware/, 2023.

[4] Blackberry, "Global Threat Intelligence Report," https://www.blackberry.com/us/en/solutions/threat-intelligence/2023/threat-intelligence-report-november, 2023.

[5] J. Bonneau, C. Herley, P. C. Van Oorschot, and F. Stajano, "Passwords and the evolution of imperfect authentication," *Communications of the ACM*, vol. 58, no. 7, pp. 78–87, 2015.

[6] J. Bonneau and S. Schechter, "Towards reliable storage of 56-bit secrets in human memory," in *23rd USENIX Security Symposium*, 2014.

[7] K. Burk, F. Pagani, C. Kruegel, and G. Vigna, "Decomperson: How humans decompile and what we can learn from it," in *31st USENIX Security Symposium*, 2022.

[8] X. d. C. de Carnavalet and M. Mannan, "Killed by proxy: Analyzing client-end tls interception software," in *Network and Distributed System Security Symposium*, 2016.

[9] J. B. Gross and M. B. Rosson, "Looking for trouble: understanding end-user security management," in *Proceedings of the Symposium on Computer Human interaction For the Management of information Technology*, 2007.

[10] H. Khan, U. Hengartner, and D. Vogel, "Evaluating attack and defense strategies for smartphone pin shoulder surfing," in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, 2018.

[11] K. Lebeck, K. Ruth, T. Kohno, and F. Roesner, "Towards security and privacy for multi-user augmented reality: Foundations with end users," in *IEEE Symposium on Security and Privacy*. IEEE, 2018.

[12] K. K. Leonard, *Encyclopedia of social measurement*. Elsevier, 2005.

[13] F. L. Lévesque, S. Chiasson, A. Somayaji, and J. M. Fernandez, "Technological and human factors of malware attacks: A computer security clinical trial approach," *ACM Transactions on Privacy and Security (TOPS)*, vol. 21, no. 4, pp. 1–30, 2018.

[14] M. Linneberg and S. Korsgaard, "Coding qualitative data: a synthesis guiding the novice," *Qualitative Research Journal*, 2019.

[15] A. Mantovani, S. Aonzo, Y. Fratantonio, and D. Balzarotti, "RE-Mind: a first look inside the mind of a reverse engineer," in *31st USENIX Security Symposium*, 2022.

[16] J. Mattei, M. McLaughlin, S. Katcher, and D. Votipka, "A qualitative evaluation of reverse engineering tool usability," in *Annual Computer Security Applications Conference*, 2022.

[17] B. Morrison, L. Coventry, and P. Briggs, "How do older adults feel about engaging with cyber-security?" *Human Behavior and Emerging Technologies*, vol. 3, no. 5, 2021.

[18] NYAN-x-CAT, https://github.com/NYAN-x-CAT/AsyncRAT-C-Sharp, 2023.

[19] petikvx, https://github.com/petikvx/LockBit-Black-Builder, 2022.

[20] S. E. Schechter, R. Dhamija, A. Ozment, and I. Fischer, "The emperor's new security indicators," in *IEEE Symposium on Security and Privacy*. IEEE, 2007.

[21] E. Spero, M. Stojmenović, Z. Hassanzadeh, S. Chiasson, and R. Biddle, "Mixed pictures: Mental models of malware," in *17th International Conference on Privacy, Security and Trust*. IEEE, 2019.

[22] StatCounter GlobalStats, "Desktop Operating System Market Share Worldwide," https://gs.statcounter.com/os-market-share/desktop/worldwide/, 2023.

[23] Virus Total, "Deception at Scale: How Malware Abuses Trust," https://assets.virustotal.com/reports/2022deception.pdf, 2022.

[24] D. Votipka, S. Rabin, K. Micinski, J. S. Foster, and M. L. Mazurek, "An observational investigation of reverse engineers' processes," in *29th USENIX Security Symposium*, 2020.

[25] R. Wash, "Folk models of home computer security," in *Proceedings of the Symposium on Usable Privacy and Security*, 2010.

[26] R. Wash and M. M. Cooper, "Who provides phishing training? facts, stories, and people like me," in *Proceedings of the CHI Conference on Human Factors in Computing Systems*, 2018.

[27] R. Wash and E. Rader, "Influencing mental models of security: a research agenda," in *Proceedings of the 2011 New Security Paradigms Workshop*, 2011.

[28] ——, "Too much knowledge? security beliefs and protective behaviors among united states internet users," in *Eleventh Symposium On Usable Privacy and Security*, 2015.

[29] wj32, "Home." [Online]. Available: https://sourceforge.net/projects/processhacker/files/stats/timeline

[30] xmrig Project, https://github.com/xmrig/xmrig, 2023.

[31] K. Yakdan, S. Dechand, E. Gerhards-Padilla, and M. Smith, "Helping johnny to analyze malware: A usability-optimized decompiler and malware analysis user study," in *IEEE Symposium on Security and Privacy*. IEEE, 2016.

[32] M. Yong Wong, M. Landen, M. Antonakakis, D. M. Blough, E. M. Redmiles, and M. Ahamad, "An inside look into the practice of malware analysis," in *ACM Conference on Computer and Communications Security*. ACM, 2021.

[33] E. Zeng, S. Mare, and F. Roesner, "End user security and privacy concerns with smart homes," in *13th Symposium on Usable Privacy and Security*, 2017.

[34] E. Zeng and F. Roesner, "Understanding and improving security and privacy in multi-user smart homes: A design exploration and in-home user study," in *28th USENIX Security Symposium*, 2019.

A Appendix

A.1 Classification Accuracy Distribution

Table 3: Classification Accuracy for each session

Session 1	***Average***	***Minimum***	***Median***	***Std. Dev.***
Basic	66.6%	50%	66.6%	12.3%
Intermediate	76.4%	50%	75%	15%
Advanced	81.9%	50%	83.3%	19.4%
Session 2	***Average***	***Minimum***	***Median***	***Std. Dev.***
Basic	79.2%	50%	83.3%	16.1%
Intermediate	79.2%	50%	83.3%	12.6%
Advanced	81.9%	50%	83.3%	16.6%

A.2 Survey Material

A.2.1 Closed Response Demographic Questions

1. What is your age?

   (a) 18–25; (b) 26–30; (c) 31–35; (d) 36–40; (e) 41–45; (f) 46–50; (g) 50+ yrs; (h) Prefer Not to Answer
2. What is your identified gender?

   (a) Male; (b) Female; (c) Non-binary; (d) Other; (e) Prefer Not To Answer
3. What is your highest level of education?
4. What is your occupation?
5. What is the approximate size of the organization you work at?

   (a) 1–49; (b) 50–249; (b) 249–999; (c) 1000–2500; (d) 2500–5000; (e) 5000+;
6. How many hours do you use your computer each workday?
7. Do you have an education in, and/or work in, the field of computer science, computer engineering or IT? (a) Yes (b) No
8. **If the answer to the previous question was yes**; How many years have you worked in the field of computer science, computer engineering, or IT?
9. Which of the following best describes your level of proficiency with technology?

   (a) None (I have very limited experience with computers);
   (b) Basic (I can perform basic tasks on a laptop/computer such as sending emails or browsing the internet);
   (c) Intermediate (I can confidently perform intermediate tasks on a laptop/computer such as changing the settings or installing new applications);
   (d) Advanced (I have knowledge of and am capable of writing source code);
10. Which of the following best describes your level of proficiency with security?

    (a) None (I have a limited understanding of security i.e., does not know what antivirus is or does not know how to use it);
    (b) Basic (I have some knowledge on aspects of security and different threats that exist and how to fix some of them);
    (c) Intermediate (I have some formal training/education/certification related to cybersecurity);
    (d) Advanced (I am a current or former cybersecurity professional or actively researches security topics)
11. **If the answer to the previous question was c or d**; How many years have you spent in training/education/certification, researching security topics or as a current/former cybersecurity professional?
12. Which operating system do you regularly use or have regularly used in the recent past (choose all that apply)? (a) Microsoft Windows; (b) MAC OS; (c) Linux or Unix

A.3 Enhanced Task Manager Interface

A.4 Software Data

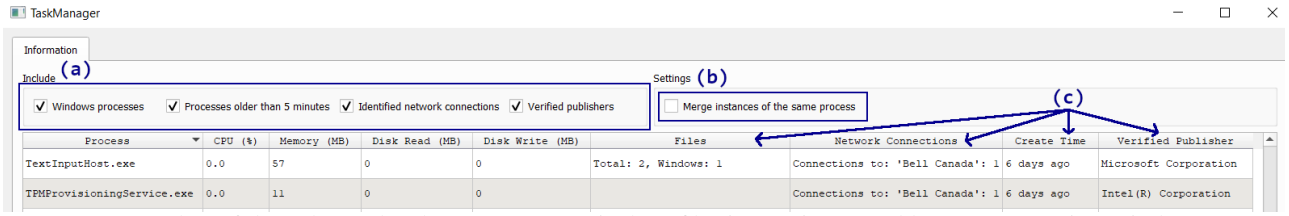


Figure 8: A screenshot of the enhanced task manager. (a) Displays filtering options to add or remove native Windows processes, processes older than five minutes, any detected network connections, or any verified publishers. (b) Allows participants to merge processes with the same name together. (c) Shows all additional information columns compared to the default Windows Task Manager (files being accessed, network connections to country/autonomous system, the create time and verified publishers).

Table 4: Benign Software File Information

File Type	*Name*	*File Version*	*Digital Signature*
Print Driver	install.exe	BRPrintAuditor Installer v∼3.0.3.0	Brothersoft Industries, Ltd.
Print Driver	jd662w632aus.exe	Brother PJ-662 v∼4.1.100.1332	Brothersoft Industries, Ltd.
File Sharing	SHAREit-KCWEB.exe	v 5.1.0.2	Smart Media4U Technology Pte. Ltd.
File Sharing	FrostWire.exe	v 4.17.2.0	None
Disk Utilities	DiskView.exe	v 2.41.0.0	Microsoft Corporation
Disk Utilities	CCleaner.exe	v 6.11.0.10455	PIRIFORM SOFTWARE LIMITED

A.5 Mitre Codes

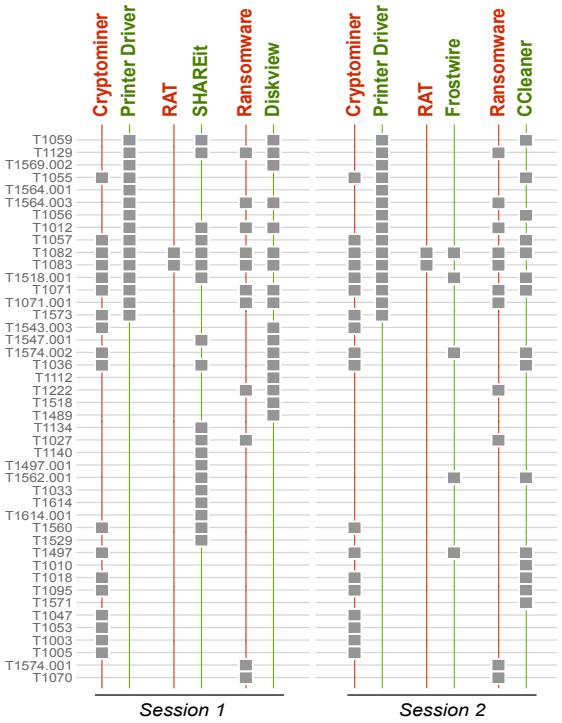


Figure 9: Mitre ATT&CK T-codes classification for the software used in the two sessions: (red) malware; (green) benign.

A.6 Coding Rules for participant observations

If the observation is related to:

- Any notification, the way the laptop or software acts Then: Program Behaviour
- The aesthetic appearance of the software, any legal documents, the aesthetic appearance of the executable Then: Program Look and Feel
- Anything related to the properties of an executable Then: Executable Properties
- Any third-party information source, or advanced technique/knowledge a participant had Then: Threat Intelligence Sources

A.7 Examples of Correct and Incorrect use of Indicators

Table 1: Codebook used to code participant comments as indicators; all examples are in context of participant verbalization (e.g. "identifying" or "ignoring" an indicator)

Code	*Correct Example*	*Incorrect Example*
Executable Properties		
Signature	Analysis of signature information, knowing that the presence of a signature is good	Trusting expired signatures, distrusting unknown signatures without analysis
Executable Metadata	Understanding useful metadata for analysis, acknowledging that some can be forged	Blindly trusting all metadata, or ignoring metadata
File Size	Understanding an acceptable file size for a given application function	Assuming larger or smaller numbers are bad or good
Publisher Reputation	Having quality expectations for a reputable company	None reported
Program Behaviour		
System Notifications	Identifying whether a prompt was giving a valid warning	Ignoring all prompts
Resource Usage	Attributing higher resource usage to more taxing processes	Ignoring resource usage, assuming all applications to be resource heavy
UI Presence	Becoming suspicious when a UI does not appear	Assuming a lack of UI is deliberate or is due to a cause they are unaware of
File Access	Questioning access to unrelated files	Assuming it is acceptable to access any file
Installation Error	Identifying why problem occurred and responding to it	Ignoring all installation errors
Customizability	Recognizing that benign programs are more likely to give installation options	Assuming customization options are a bad sign
Execution Latency	Noticing the delay in running the program and questioning whether it should happen or not	Assuming that a delay is typical behaviour
Informativeness	Recognizing when an application gives little information	Not paying attention to any installation status
Program Look and Feel		
Aesthetics	Recognizing inconsistent graphics with known organizations interfaces	Identifying the presence of pictures as an indicator of benign software
File name	Recognizing spoofed names, knowing names can easily be changed	Assuming numbers, capital letters, and symbols are automatically bad or good
Typos and Grammar	Identifying spelling errors or sentence structure problems	Spotting grammatical issues but disregarding them
Icon	Identifying the lack of an icon or identifying masquerading application icons	Assuming the presence of an icon is good, or that the default icon is the application's icon
Social Media Links	Identifying that links can be copied, using them to verify software information	Assuming the websites and applications were made by the same people
Eula/Legal document	Identifying if documents reference the correct country, acknowledging they can be copied	Assuming EULA/Legal Documents are the best way to evaluate a software
File extension	Acknowledging that a ".exe" is an executable	Assuming a ".exe" means it is either safer or more dangerous
Threat Intel. Sources		
Online Search	Finding relevant data online to help with an analysis	Searching for the publisher information or trusting unverified sources of information
Prior Knowledge	Remembering previous experiences or knowledge that aid in analysis	Confusing current application with unrelated past experiences
Network Connections	Viewing network connections to help determine validity of applications	None reported
Online Scanning	Usage of online tools to verify the binary of a software	None reported
Hash Lookup	Comparing file hashes with legitimate applications	None reported
DNS	Using DNS information to track connections	None reported