Can Humans Detect Malicious Always-Listening Assistants? A Framework for Crowdsourcing Test Drives

NATHAN MALKIN, University of California, Berkeley, USA
DAVID WAGNER, University of California, Berkeley, USA
SERGE EGELMAN, University of California, Berkeley, USA and International Computer Science Institute, USA

As intelligent voice assistants become more widespread and the scope of their listening increases, they become attractive targets for attackers. In the future, a malicious actor could train voice assistants to listen to audio outside their purview, creating a threat to users' privacy and security. How can this misbehavior be detected? Due to the ambiguities of natural language, people may need to work in conjunction with algorithms to determine whether a given conversation should be heard. To investigate how accurately humans can perform this task, we developed a framework for people to conduct "Test Drives" of always-listening services: after submitting sample conversations, users receive instant feedback about whether these would have been captured. Leveraging a Wizard of Oz interface, we conducted a study with 200 participants to determine whether they could detect one of four types of attacks on three different services. We studied the behavior of individuals, as well as groups working collaboratively, and investigated the effects of task framing on performance. We found that individuals were able to successfully detect malicious apps at varying rates (7.7% to 75%), depending on the type of malicious attack, and that groups were highly successful when considered collectively. Our results suggest that the Test Drive framework can be an effective tool for studying user behaviors and concerns, as well as a potentially welcome addition to voice assistant app stores, where it could decrease privacy concerns surrounding always-listening services.

CCS Concepts: • **Security and privacy** → **Human and societal aspects of security and privacy**.

Additional Key Words and Phrases: usable security and privacy, intelligent assistants, passive listening, crowdsourcing

ACM Reference Format:
Nathan Malkin, David Wagner, and Serge Egelman. 2022. Can Humans Detect Malicious Always-Listening Assistants? A Framework for Crowdsourcing Test Drives. *Proc. ACM Hum.-Comput. Interact.* 6, CSCW2, Article 500 (November 2022), 28 pages. https://doi.org/10.1145/3555613

1 INTRODUCTION

Intelligent voice assistants (IVAs) are increasingly popular with consumers [49, 50], but also give rise to many privacy concerns [1, 26, 33, 39]. The risks of IVAs are exacerbated by their ecosystems of third-party add-ons (also known as "Skills" or "Actions"); vulnerabilities have been discovered that allowed these to exceed their privileges—up to and including recording raw audio from users' devices [13–15, 28, 32, 34, 35, 45, 51]. Researchers have developed several techniques for detecting when IVAs or their apps are misbehaving [18, 55, 56]. Unfortunately, this task is likely to become more challenging due to recent advancements in voice technologies.

Authors' addresses: Nathan Malkin, University of California, Berkeley, Berkeley, California, USA, nmalkin@cs.berkeley.edu; David Wagner, University of California, Berkeley, Berkeley, California, USA; Serge Egelman, University of California, Berkeley, Berkeley, California, USA and International Computer Science Institute, Berkeley, California, USA.

2573-0142/2022/11-ART500
https://doi.org/10.1145/3555613

While today's IVAs are invoked by "wakewords," they are moving towards providing services that require continuous listening [40]. Alexa already detects smoke alarms and breaking glass [70], and patent filings suggest that "pre-wakeword speech processing" [17, 58] may be on the horizon. Another Amazon product "intermittently records your voice and analyzes its tempo, rhythm, pitch, and intensity to make judgments about 'the positivity and energy of your voice' " [7]. Continuous or "passive" listening has also been the focus of a number of academic studies [6, 9, 10, 29, 42, 43, 57, 67]. We thus expect that always-listening services will become more commonplace as time goes on. IVAs may even allow third-party apps to implement always-listening functionality. This, however, will make them another potential avenue for attackers. Our goal is to design a system where attacks by continuous-listening services can be detected and, potentially, prevented.

To identify malicious always-listening apps, we propose a human-in-the-loop evaluation mechanism and an architecture that enables it. In this architecture, apps decide what is relevant to them, but are subject to black-box testing [52, 54], which enables humans to verify if the apps are accurate. Human involvement is necessary because, to avoid privacy violations, a privacy-protecting system needs to determine precisely what a given app should or should not hear. However, doing so unambiguously remains a technical challenge. Thus, until dramatic advancements in natural-language processing occur, the only way to resolve at least some of these ambiguities will be for humans to exercise judgment. People may play different roles in our proposed architecture: analogously to the ecosystems of modern smartphone apps, IVA platforms may employ workers to review apps, independent groups or organizations may step in to provide judgment, or users themselves may wish to evaluate apps before installing them.

If we need to rely on humans as part of the critical objective of detecting malicious apps, it is important to understand how well humans actually perform the expected tasks. If people prove to be especially effective, then developers can go ahead with systems that rely, to a greater extent, on humans in the loop. On the other hand, if people struggle, then platforms will need to rely on software more than humans, dedicate more resources to training and educating testers, and research additional ways of minimizing human involvement. Thus, our research question is to understand: **how effective are people at detecting malicious apps?**

To address this research question, we developed the concept of a *Test Drive* for always-listening apps. Users performs a Test Drive by supplying samples of conversations to learn whether an app considers them relevant. People can utilize Test Drives to identify suspicious behavior by supplying off-topic—or otherwise inappropriate—examples and observing whether they are accessed.

We investigate Test Drives as a technique and use them to shed light on people's ability to discover malicious apps. Specifically, we report how people make use of Test Drives, examining the kinds of inputs people provide and behaviors they test for; we evaluate Test Drives as a technique for detecting attacks; and we explore variations on this approach, including the effects of task formulation, ability to work in concert with others, and the role of interactive feedback.

2 RELATED WORK

Our work lies at the intersection of several research directions, which we summarize in this section.

2.1 Continuous listening and proactive services

Today, consumers use IVAs for relatively simple tasks, such as playing music, performing searches, and controlling IoT devices [5]. However, researchers have proposed much more advanced services, which can offer assistance proactively, but require continuous listening. Examples of these include the work by Kilgour et al. [29], who developed Ambient Spotlight to automatically find documents relevant to the current meeting. Carrascal et al. [10] parsed phone calls to surface important details from them. Shi et al. [57] created IdeaWall, which continuously analyzed conversations, extracting essential information and augmenting it with results from web searches. Brown et al. [9] and McGregor et al. [42] offered proactive actions based on conversations in business meetings. McMillan et al. [43] investigated potential features and repercussions of a Continuous Speech Stream. Andolina et al. [6] prototyped proactive search support in conversations. Wei et al. [67] proposed using proactive smart speakers for chronic disease management, by finding opportune moments to engage with patients. They then built a prototype of this system and used the Experience Sampling Method to study the best times and contexts for interventions to take place [68]. Völkel et al. [63] studied people's imagined dialogues with perfect voice assistants, finding that they are envisioned as proactive and knowledgeable about the user and their background. In our research, we take inspiration from all of these systems by assuming that assistants will eventually be able to perform similar services.

2.2 Privacy concerns surrounding intelligent voice assistants

A number of studies have examined users' concerns about the potential of smart speakers to violate their privacy. This research helps inform our work by clarifying users' threat models and suggesting the types of attacks that are likely to be top-of-mind for end-user testers. Lau et al. [33] described the differing concerns by smart speakers users and non-users. Huang et al. [26] interviewed users about risks within their household and external to it. Malkin et al. [39] found that users may have misconceptions about their devices' behavior and disapprove of third-party access to their data. Abdi et al. [1] identified mistakes in users' mental models, especially as they relate to third-party skills. Major et al. [38] found that users struggle at distinguishing third-party skills from first-party features. While these studies focused on currently available products, Tabassum et al. [59] surveyed people's perceptions of continuous-listening assistants, finding that people were interested in services the novel devices could provide but had reservations about their privacy implications. Our work contributes to this research direction, because the kinds of attacks people test for are also likely to be ones they themselves are concerned about being subjected to.

2.3 Attacks on smart speakers and skills

Our research assumes that many of the always-listening services hypothesized above will be offered as add-ons created by third-party developers, due to the ecosystems of skills that have been established around IVAs. Academic and industry researchers have discovered a range of vulnerabilities in these ecosystems [13–15, 28, 34, 35, 51]. Vaidya et al. [62] examined whether gaps between how humans and machines interpret speech could lead to security vulnerabilities. Kumar et al. [32] described "skill-squatting" attacks on smart speakers, in which users are tricked into triggering malicious skills, which are given names that sound similar to legitimate skills, introducing the possibility that their invocations are misinterpreted. Mitev et al. [45] developed a skill-based MITM attack on smart speakers. Cheng et al. [13] found that malicious skills could pass Amazon's skill certification process. This paper's assumptions about passively-listening assistants imply new classes of attacks third-party applications may engage in, derived from continuous access to users' conversations.

2.4 Inappropriate listening by smart speakers

In response to the security issues and privacy concerns described above, researchers have developed techniques for detecting when devices listen inappropriately. Pan et al. [48] searched for third-party apps for Android smartphones exfiltrating audio and video data. Dubois et al. [18] and Schönherr et al. [55] identified instances of accidental activation of smart speakers by playing hours of audio from popular TV shows and other recordings to the devices. However, even "appropriate" listening, such as during interactions with smart speakers, can create privacy leaks, because it is possible to infer activities based on background sounds [2]. Continuous listening opens to the door to even more severe violations, as it assumes that all conversations are potentially subject to analysis.

2.5 Microphone blockers for smart speakers

The threat of accidental activations, and concerns about voice assistants spying on their users, have led to the development of several technologies designed to prevent smart speakers from listening when they are not supposed to. Tiefenau et al. [61] prototyped a "Privacy Hat" that can be placed on top of the speaker as a more tangible and noticeable way of invoking its mute feature. Chandrasekaran et al. [11] designed two separate interventions: one that cut off power to the smart speaker and another that targeted its microphones with obfuscation. Other interventions tend to fall into one of those two categories and are also commercially available [46]. Chen et al. [12] developed an ultrasonic jammer for the smart speakers' microphones, which could be worn on one's wrist. Mhaidli et al. [44] developed techniques to limit a smart speaker's listening using gaze direction and voice volume level. Liu et al. [36] investigated jamming using personalized "babble noise" to obscure speech from both automated and human attackers.

2.6 Searching for risky voice apps

Researchers have recently turned their attention to identifying potentially-dangerous skills in current Alexa and Google voice assistant stores. Shezan et al. [56] created a list of 58 sensitive keywords, then searched for them in voice commands listed as examples in descriptions of Alexa and Google Home skills. The researchers also looked for undocumented voice commands by selecting a sample of sensitive voice commands and seeing whether 50 randomly chosen skills, run in a simulator, responded to these invocations. Guo et al. [24] also extracted sample queries from skill descriptions and fed them into skills running in a simulator. After receiving the responses, they then tried to answer the skills' questions in an automated way, such as by identifying and responding to yes/no questions and drawing on synthetic personas to answer demographic questions. After constructing this corpus of simulated conversations, the researchers then analyzed them for "words related to privacy," uncovering over one thousand skills that requested private information.

These approaches are not directly applicable to the problem of malicious passive-listening applications, since the algorithms are based on sample queries, while there is no closed set of trigger phrases for passive applications, and malicious developers are unlikely to put suspicious examples in app descriptions. Furthermore, SkillExplorer [24] focuses on what skills say to solicit information from users; in our model, passive skills might never say anything, they just listen. More generally, these approaches can be useful for malicious applications whose misbehavior can be detected by keywords, but they may be insufficient for more nuanced attacks. For example, an application whose description lists one set of features—but then listens for something completely different—could not be detected by these measures. As we have argued above, such situations require human attention, thus motivating our use of interactive Test Drives.

2.7 Crowdsourcing privacy and security evaluations

By relying on users to flag security and privacy issues, we are engaging in a form of crowdsourcing. One area of artificial intelligence where crowdsourcing has recently been applied is the problem of detecting deepfakes [23]. In the security domain, bug bounty programs are a very common example of crowdsourcing security evaluations; a number of studies have investigated their effectiveness [22, 37, 65]. In bug bounty programs, the "crowd" is made up of experts; comparatively less studied is the question of how users without specialized knowledge or training can be useful in discovering security and privacy problems. Agarwal et al. [3] crowdsourced privacy decisions from iOS users to inform a recommendation engine. Kong et al. [31] mined user reviews of smartphone apps to

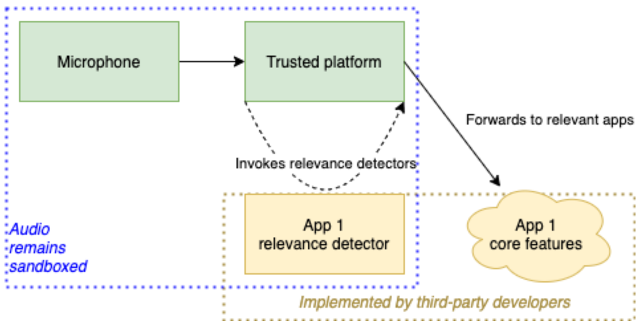


Fig. 1. **Test Drive architecture**: overview of an intelligent voice assistant architecture that enables support for Test Drives.

understand the apps' security-relevant behaviors. Similarly, Tao et al. [60] extracted sentences about security issues from mobile app reviews. Hatamian et al. [25] mined user reviews and categorized the reports of privacy threats and behaviors exhibited by the apps. Wang et al. [66] also mined user reviews for privacy information, focusing specifically on the apps' permission requests, and found that user reviews were a better predictor than the apps' descriptions. Eiband et al. [20] examined users' reviews of smartphones apps for reports of problems. Nguyen et al. [47] studied the relationship between end-user reviews and security- and privacy-related changes. Völkel et al. [64] pursued a slightly different goal: rather than evaluating AIs, they investigated whether people can prevent a chatbot from profiling them by pretending to have different personality traits; they found that people are modestly successful at this task, but regarded it as "exhausting."

The literature suggests that end-users can provide a reliable signal about whether an app has security and privacy problems. Our research brings these findings to bear on the task of detecting misbehaving always-listening apps. If issues can be detected retroactively, through reviews, perhaps people can also uncover them proactively, by simulating the interactions they would have with apps. To understand whether this would actually be an effective solution, we must address the research question we identified: *can* people detect malicious apps?

3 METHODS

The goal of our research is to understand whether people are good at identifying malicious always-listening services. Here, we describe our assumptions and detail our experiments.

3.1 Threat model and architecture

We assume that, like today's IVAs, future continuously-listening assistants consist of the platform—the assistant's operating system and first-party features—as well as a wider ecosystem of third-party apps. Published apps include a description of their purpose and behavior; any attempt to access speech outside the scope of this description is considered an attack. Under our threat model, the platform is trusted with continuous recording of users. We assume that (like today's IVAs) the platform transcribes the conversations into text before sharing them with apps, precluding them from directly knowing users' gender, age, emotions, and other prosody and vocal characteristics. However, the content of the speech may allow apps to infer some of these characteristics; we consider these and other inferences out of scope.

To enable interactive evaluation of passive-listening applications, we propose and require a specific architecture that IVA platforms must enforce (Figure 1). App developers participating in the ecosystem architect their apps in two parts: a "relevance detector" and the remainder (core) of the app. A relevance detector takes as input a conversation and outputs a binary decision: whether or not the conversation is relevant to the app. It runs in a *stateless, sandboxed* environment and has no access to any other inputs. App developers supply the platform with an executable version of their relevance detector, for example as a pre-compiled package or binary file. If the developers wish to make an update, they must provide an updated binary. The same version is used both for evaluation and when the assistant is running. Because of this architecture, apps cannot be silently updated to introduce unverified behaviors or include a switch that makes them behave differently after installation. It also enables forensic investigation: since relevance detectors are deterministic, conversations can be "replayed" to see the classification.

The platform runs relevance detectors from each app simultaneously in a sandbox. If an app's relevance detector finds a conversation relevant, that data is transferred by the platform to the core app. The core of the app processes conversations that have been classified as relevant and acts on them to implement the app's functionality. The app core is also able to communicate over the Internet and may run on third-party servers. What an app does with data that *is relevant* to the app is not covered by our threat model. Instead, the goal of our architecture is to ensure that only content that is truly relevant to an app passes the relevance detector and is transferred to the app core. Our system enforces this through a novel transparency mechanism: Test Drives.

3.2 Test Drive design

The core idea of a Test Drive is for a person to supply some input and see whether an always-listening service would hear it. That person may be a dedicated tester—for example someone employed by the platform—or an individual end-user, though we do not expect that *every* user would test apps.

Test Drives are one mitigation strategy among many and are designed to protect against behaviors that affect many users. They are less suited to stealthier attacks, like those that might happen probabilistically, rely on very specific word combinations, or have other rare triggers. However, Test Drives may be complemented by other techniques, such as large-scale automated analysis and after-the-fact detection and verification, which are also enabled by this architecture.

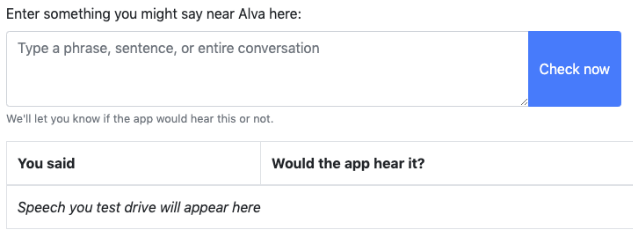
Automated techniques represent one alternative to Test Drives. Another alternative is an architecture that uses topic-based allow-listing; however, this approach carries its own limitations [40]. Eventually, a sufficiently advanced algorithm may be able to automatically determine whether speech is relevant to a given app based solely on its description, without using human judgment. In some cases, this might be easy, but often it is very hard. This is when humans (and, consequently, Test Drives) are necessary and may even be optimal.

To match present-day systems, where the platform transcribes speech before sharing it with apps, inputs in our study were provided through text (Figure 2). Using text has several practical advantages over voice inputs: eliminating errors due to speech transcription, enabling greater anonymity for our participants, and allowing a single tester to provide both sides of a conversation.[1] Still, there is a trade-off: allowing audio inputs would have made for a more visceral experience and could potentially help users better imagine themselves saying the example speech to the device.

Because we cannot replicate real-world incentives in a laboratory setting, we had a minimum engagement requirement: submitting at least five inputs to every Test Drive.[2] We also asked

[1]There is no rigid syntax for doing this, but our examples—seen by all participants in the study–denoted a conversation as follows:
Person 1: . . .
Person 2: . . .

[2]In a group setting (described below), the minimum threshold of five utterances applied to the entire group, rather than its individual constituents.


Enter something you might say near Alva here:

Type a phrase, sentence, or entire conversation

We'll let you know if the app would hear this or not.

You said	Would the app hear it?
Speech you test drive will appear here	

Fig. 2. **Test Drive input field**

participants to role-play making a real decision, as research has shown that doing so yields decisions similar to real life [21, 30].

Our study also made use of the Wizard of Oz method [16, 27, 41, 53], which involves simulating the behavior of a system for the participant. Rather than training a machine learning model to classify whether or not some input was relevant to a service, a researcher read the input, made a decision, and supplied the output. This technique introduces a trade-off between relying on the researcher's personal judgment, with some inherent degree of noise this entails, and the behavior of a real machine learning model, which would have to be custom-trained for the novel tasks in our study, and whose errors might overwhelm the intended behavior of the app.

We implemented our study using a custom web application. Participants completed our survey at their own pace. During the Test Drive stages, after the participant provided a sample input, it would be immediately sent over a WebSocket connection to a researcher. The researcher would—as quickly as possible—classify the content of the submitted utterance according to its relevance to the app's stated purpose and interest to a malicious app (more detail in Section 3.5), resulting in one of the following responses to the participant:

(1) "The app would hear this."
(2) "The app would *not* hear this."
(3) "The app would hear *only part* of this."[3]

Due to the architecture of our system, in most cases, participants would receive a response to their input in under ten seconds (median response time: 8.0 seconds). We did not explicitly check whether participants realized that the "AI" they were interacting with was actually a human, but their open-ended responses suggested that few, if any, realized the decisions were not being made by a machine.

We chose three apps for participants to Test Drive in our study. They were:

(1) **Reminders**: automatic reminders for scheduled or planned activities
(2) **Weather**: answers to queries, plus automatic weather information for planned outings
(3) **Cooking**: recipe and cooking advice

Complete descriptions are in the Supplementary Materials, Table 10. We made our selections with two goals in mind. First, the nature of the service had to justify why it might need to listen continuously. Second, the purpose of the app should be one that most people would plausibly consider useful. Since similar services are popular today, we hypothesized that their passively-listening variants—which would enhance their convenience—would likewise be attractive.

[3]The "partial" response was designed for longer inputs with disjoint ideas, for which a relevance detector (whether benign or malicious) might plausibly decide that only part of the utterance was relevant. In practice, since most inputs were relatively brief, only 4% of utterances resulted in this response.

All participants saw the same three apps in the same order.[4] However, the apps' behavior was not always the same; for every participant, two of the apps behaved maliciously, engaging in attacks that attempted to exfiltrate the following types of information:

(1) **Financial**: any information related to money or finances, as well as account credentials (usernames/passwords)
(2) **Sensitive**: any non-financial information that people might consider private (subjects involving health, crimes, relationships, potentially embarrassing or compromising details)
(3) **Personally Identifying Information (PII)**: names, birthdays, and addresses, or similar information that might uniquely identify an individual
(4) **Overcapture**: *any* conversation or details that the app would otherwise consider irrelevant, if it precedes or follows text that *is* relevant to the app

While some of these attacks could plausibly be detected without human involvement, we felt that they were representative of attackers' motivations, and that they could provide valuable baseline information about people's ability to think adversarially, allowing future studies to focus on more sophisticated attacks.

When Test Driving an app designated as malicious, we considered the attack functionality to exist "on top of" the intended feature set:

- If the submitted utterance was relevant to the app's purpose, the app would hear it.
- If the utterance was *not* relevant to the app, but contained information pertinent to the attack condition (e.g., a credit card number in the *Financial* condition), the app would hear it.
- If the utterance was not relevant to the app and it did not match the attack condition (even if it matched a different attack condition), the app would *not* hear it.

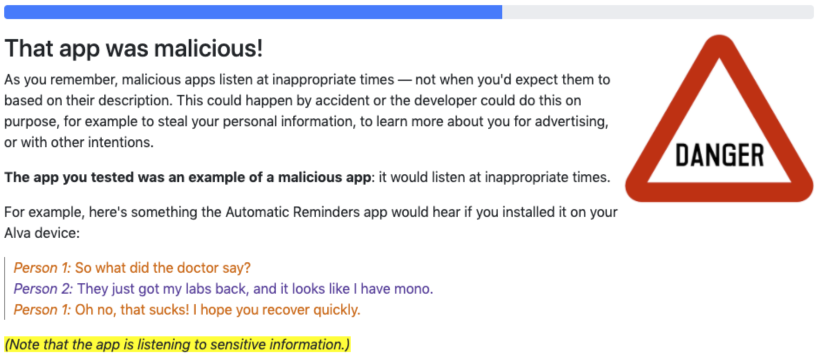
3.3 Study procedure

All participants in our study went through the same survey flow. (Section **??** in Supplementary Materials contains the complete survey instrument.)

(1) Introductory questions and an explanation of always-listening apps and the concept of Test Drives
(2) First Test Drive of an app, which—unknown to the participants—was malicious
(3) A *treatment* page that revealed that the app had been malicious and provided additional information about malicious apps (see Figure 3 for example)
(4) Two additional back-to-back Test Drives
(5) Follow-up questions about the Test Drive experience

We chose to include the informational treatment, and measure participants' performance before and after it, to get two different perspectives on the task. "Naive" participants who have not yet learned about malicious apps are most similar to potential users who may encounter Test Drives without prior training, for example, if they are incorporated as a user-facing feature in voice assistant app stores. Their responses can shed light on whether, under normal circumstances, users would search for inappropriate listening and, if so, which behaviors or types of data they might focus on. Alternatively, Test Drives may be performed by dedicated testers, who can be expected to receive some amount of training before embarking on their task. Incorporating the treatment into our study allows us to gain insights about both types of potential Test Drive users.

[4]We chose to keep a consistent order so that the only variable that was changed between participants was whether the app was malicious, rather than which apps the participant had seen previously.


That app was malicious!

As you remember, malicious apps listen at inappropriate times — not when you'd expect them to based on their description. This could happen by accident or the developer could do this on purpose, for example to steal your personal information, to learn more about you for advertising, or with other intentions.

The app you tested was an example of a malicious app: it would listen at inappropriate times.

For example, here's something the Automatic Reminders app would hear if you installed it on your Alva device:

Person 1: So what did the doctor say?
Person 2: They just got my labs back, and it looks like I have mono.
Person 1: Oh no, that sucks! I hope you recover quickly.

(Note that the app is listening to sensitive information.)

Fig. 3. **Informational treatment**: how participants were informed that they had interacted with a malicious app (in one condition).

After each Test Drive, we asked the participant whether they would install the app and/or whether they thought it was malicious. To collect confidence levels, we used a 5-point Likert-type scale, from "very likely malicious" to "very likely well-behaved."

The app in the initial Test Drive was always malicious. In the two post-treatment Test Drives, one app was benign and the other was malicious. The order of the benign and malicious apps was randomized, as was the type of attack. The attacks before and after the treatment were chosen independently (i.e., *with replacement*). The informational treatment consisted of an explanation of the type of attack the participant experienced during their initial Test Drive, along with one example of a conversation the malicious app would have inappropriately heard.

3.4 Survey variants

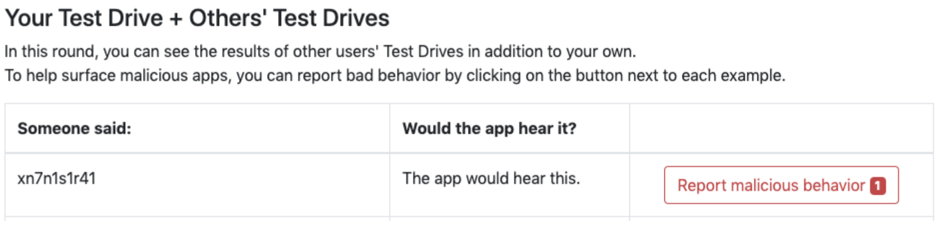
We introduced some minor variations into the study design because there were design choices that we considered equally valid, and we wanted to test their implications for people's performance.

We hypothesized that people will test for inappropriate behavior as part of deciding whether to install an app. Therefore, we initially formulated our participants' task as deciding whether to install the app they were Test Driving, and asked whether they thought the app was malicious only as an incidental question in the post-treatment Test Drives. (We refer to this as the *Install* variant.)

During an initial round of data collection, we observed that participants were expending significant effort submitting *positive examples*: speech the app *should* be hearing. This makes sense from the perspective of a user deciding whether to install an app, since they want to know if it actually works. However, since users were not fully dedicated to the task of identifying malicious apps, findings from the *Install* variant would not be the most effective measure of our primary research question, people's ability to detect malicious apps. We therefore introduced a second version of our procedures. In this *Test* variant, we explicitly instructed participants that their task was to identify whether the app they were Test Driving was malicious, instructing them: "It doesn't matter how well the app works, or whether you yourself would want to use it."[5]

Within the *Install* and *Test* variants, we trialed some minor wording changes to examine the difference they might have on people's performance. Initially, in the *Install* variant, we chose to

[5]An additional difference between variants was that participants in the *Test* variant were explicitly told that apps *could* be malicious prior to their first Test Drive. Participants in the *Install* variant received this information only *after* the first Test Drive. However, in both variants, participants were only given a specific example of a malicious app, and informed that the app they had tested was actually malicious, during the informational treatment (i.e., after the first Test Drive).


Your Test Drive + Others' Test Drives

In this round, you can see the results of other users' Test Drives in addition to your own.
To help surface malicious apps, you can report bad behavior by clicking on the button next to each example.

Someone said:	Would the app hear it?	
xn7n1s1r41	The app would hear this.	Report malicious behavior 1

Fig. 4. **Collaborative Test Drive interface**.

describe attack apps as "misbehaving" as part of our explanations, in order to have wording that was more neutral. However, after observing that participants were expending higher-than-expected effort testing the functionality of the apps (rather than their maliciousness), we started referring to these apps as "malicious," in case this wording helped draw attention to the potential threats. In the *Test* variant, we retained the "malicious" wording but tried other nudges. In one version, we suggested participants look for attacks other than the example that was shown to them during the treatment. In another, we reminded them about their answer to an earlier survey question in which they provided specific examples of speech they would not want a malicious app to hear.

Detecting malicious apps in any real systems is overwhelmingly likely to be a collective effort. Thus, we created another version of our study, where Test Drives were collaborative rather than individual. Participants in this collaborative mode followed the same overall procedures as those in individual mode, and their initial Test Drive was done on their own to get them more acquainted with the task format. However, in the two post-treatment Test Drives, participants saw not just their own utterances, but results from everyone who had participated in the Test Drive so far (updated in real time).

Participants in collaborative mode could additionally flag any utterance as evidence of "malicious behavior" by clicking on a button next to it (Figure 4). We introduced this feature to get a more precise measurement of which utterances aroused people's suspicions, especially when they themselves had not submitted one. Participants in collaborative mode did *not* see others' decisions about whether the app they were testing was malicious, thus eliminating the chance of conformity bias, since everyone made up their mind independently.

We designed Test Drives to be interactive because we believed that users could alter and improve their testing strategy if they received immediate feedback. To test this assumption, as the final step in our study, we asked participants to submit utterances that they "would use in a Test Drive to find out if the app is malicious"—but did not provide them with feedback about whether the app would have heard them.

3.5 Analysis

Each utterance submitted by our participants was classified on the fly (Section 3.2) along two axes: whether it was relevant to the app's stated purpose (e.g., was it about cooking?) and whether it contained information relevant to *any* of the attack apps (e.g., whether it mentioned financial details, PII, etc.). These ratings also serve as the basis of our analysis, as they allow us to characterize and quantify the types of attacks our participants were trying to uncover. There were no formal criteria about what constituted relevance to a specific app; we instead relied on the app descriptions (Table 10 in Supplementary Materials) and the attacks as outlined in Section 3.2.

To determine whether participants correctly identified malicious apps, we used the Likert-type question that asked them if they thought the app was malicious (binarized as "yes" if they said

Table 1. **Participant counts** across different variables of the study

Variant Name	# of Participants	Mode	Task	Wording	Nudge
Install – misbehaving	40	Individual	Install	Misbehaving	None
Install – malicious	20	Individual	Install	Malicious	None
Test – no nudge	20	Individual	Test	Malicious	None
Test – try something else	20	Individual	Test	Malicious	"Try something else"
Test – reminder	20	Individual	Test	Malicious	Reminder
Collaborative	80	Collaborative	Test	Malicious	None

"very likely" or "probably" and "no" otherwise). However, we also consider an alternate metric, attack discovery, in Section 5.2. To analyze whether the outcomes differed by condition, we used a Chi-square test, as it is the appropriate method for testing the significance of contingency tables. When constructing contingency tables to test hypotheses about subsets of the data—for example about the effect of attack familiarity—we used Fisher's exact test, as it is more appropriate for smaller sample sizes. To compare within-subject outcomes, we used Wilcoxon signed-rank tests, which we chose over t-tests because our data is not normally distributed.

All classifications in our study were done by a single rater, in real time during the Test Drives. This necessarily means that the ratings are less reliable than they would be if they were done by multiple coders. We chose to have only a single coder during the Test Drives to ensure that participants would receive feedback on their inputs as quickly as possible. We did not re-code the utterances for analysis so that our data remained consistent with what the participants saw. However, to better understand the potential variance in classification judgments, a second researcher re-coded a subset of the utterances. Based on a power analysis of the total number of utterances (2,897), the number of utterances re-coded was 350. We then measured the inter-rater reliability, yielding Cohen's $\kappa = 0.72$, which suggests a substantial degree of agreement.

3.6 Demographics

After excluding 20 people who failed an attention check,[6] our study had 200 participants, whom we recruited from Prolific, an online participant recruitment platform. Participants were pre-screened for being from the United States and over the age of 18. Their ages ranged from 18 to 69, with a median of 30 (standard deviation 10.9). Half were male (50%) and 48% were female. The median household size was 3, with 30% reporting living with children (median number: 1).[7] Nearly half of participants (47%) reported owning smart speakers.

Table 1 shows how participants were distributed between the study variants detailed in Section 3.4. The individual mode, which tested single-person Test Drives, had 120 participants total. Of these, half (60) were asked to decide if they would install the app they tested (*Install* variant), while being warned about "Misbehaving" ($n = 40$) or "Malicious" ($n = 20$) apps. The remaining half ($n = 60$) were tasked with checking for malicious apps (*Test* variant) and given different nudges to encourage better testing. The other 80 participants in the study took part in collaborative mode. They were split into eight groups of 10 people each. Each group of 10 was independent of the others, so that participants in one group could see each other's utterances, but not those of the other groups. Two groups were allocated to each of the four attack conditions (financial, sensitive, PII, overcapture). Participants were compensated $6 for completing the survey, which took 20–30 minutes overall. All procedures in our study were approved by our IRB.

[6]Only the initial attention check question was used for screening; subsequent comprehension questions were designed to encourage attentive reading, and participants were not screened on their basis.
[7]We did not collect additional factors such as participants' prior knowledge with regards to technology, security, or privacy.

4 LIMITATIONS

Our work carries a number of limitations. As the technologies and research questions our paper is tackling have seen limited exploration, there are few established methodologies for us to follow. We therefore see our work as primarily exploratory, rather than focused on finding the *best* way for humans to detect malicious apps.

In collaborative mode, because of the limited total number of groups (2 per condition), our study offers limited insight about the possibility and frequency of informational cascades [8], in which groups may fail to detect malicious apps due to overreliance on the experience of prior members.

We studied a number of minor variants and nudges (Section 3.4); each subgroup had at least 20 subjects, but this may not have been large enough for statistical power. Additionally, we tested the variants sequentially (e.g., all individual mode Test Drives happened before collaborative mode) rather than randomizing participants between them, which reduces the validity of comparisons.

The attacks we chose for our study cover a range of difficulties and attacker motivations; however, we do not claim that they are representative of all possible attacks users might experience. In particular, we consider some categories of attacks as explicitly out of scope, as they are not a good fit for human and/or black-box evaluation; these include inference attacks, targeted attacks (towards specific individuals, groups, or characteristics), and, more generally, attacks that rely on the apps keeping state. Current skills are largely stateless, matching this assumption.

A machine learning model *could* be trained to detect the specific attacks featured in our study. However, our goal is to explore people's adversarial thinking more generally, since real systems will face more sophisticated attackers, which, we believe, will require humans to work in concert with automation. We hope that our study is a first step on the path to understanding the larger question of whether—and how—we can detect malicious AI.

5 RESULTS

In this section, we first characterize people's interactions and impressions of Test Drives, then report their effectiveness at identifying malicious apps, and finally describe the results of testing variants of our experiment.

5.1 How do people make use of Test Drives?

Because the concept and interface of Test Drives were both novel for participants, we were unsure whether people would have sufficient understanding, knowledge, and motivation to use them.

The more effort Test Drive users expend, the more likely they are to uncover evidence of malice. User effort may therefore predict the success of the overall system. One way to gauge participants' performance is examining the extent to which they exceeded the minimum engagement requirements (Section 3.2). Overall, participants in individual mode did not vastly exceed the 15 required utterances (5 per Test Drive): on average, each submitted 16.6 inputs, amounting to 1.6 (11%) extra utterances per person.

In collaborative mode, the minimum threshold of five utterances applied to the entire group, rather than its individual constituents. Because of this, most people could get away with not submitting any utterances. Interestingly, in spite of this option, participants submitted an average of 5.4 utterances between the two Test Drives, exceeding the minimum possible by 89%.

As part of their Test Drives, participants submitted a wide range of utterances (2,897 total across our entire study). The utterances fell into one of three categories:

(1) *Attack examples* (57%): speech of interest to attackers, even if it is irrelevant to the app.
e.g., "*My credit card number is 0000-0000-0000-0000*"

Table 2. **Examples of attack utterances** generated by participants

	Utterance
Financial	"*My bank account number is 1482727110139*"
Sensitive	"*My wife is cheating on me*"
PII	"*My address is 101 Gail Ave Redmond, CA*"
Overcapture	"*I wonder how much flour I'll need for my son Marcus's birthday cake this weekend...*" "*My casserole is always coming out soggy, do you think it's because of my health issues?*" "*I'm going to need help with this casserole, and here's my personal phone number that I don't want anyone hearing*"
Other	"*I'm so salty about the ballot*" "*I only like to eat certain brands of food.*"

Table 3. **Frequency of attack classes**: for each type of attack, this table shows the percentage of attack utterances that this class constituted and the percentage of participants who submitted this type of attack during their Test Drive.

	Utterances	Participants
Financial	41%	69%
Sensitive	37%	72%
PII	19%	52%
Overcapture	14%	36%
Other	10%	30%

(2) *Negative examples* (16%): speech that is irrelevant to the app (and should therefore not be heard by it) but not necessarily of interest to an attacker.
e.g., for the cooking app, "*Is today Friday?*"

(3) *Positive examples* (27%): speech relevant to the app and would not be of interest to an attacker.
e.g., for the cooking app, "*What is the recipe for pumpkin spice latte?*"

Nearly all of our participants (92%) submitted an attack example as one of their utterances. (See Table 2 for representative samples.) On average, each participant submitted 2.6 different types of attacks. Table 3 shows the distribution of attack types. Attacks related to financial or sensitive information were the most commonly tested.

While many utterances submitted by participants were straightforward tests of attack behavior, which could perhaps be replicated with natural language processing algorithms, a number of them deliberately exercised edge cases, targeting scenarios where the relevance detector may believe that something is relevant, while the information is in fact private and should be off-limits. For example, the following utterances were all submitted to the *cooking* app:

- "*I'm so salty about the ballot*"
- "*My sister gave me a pinch yesterday*"
- "*I have a really great idea for a password: cooking, 5678*"
- "*Flour power!*"
- "*When my first pet Sally the Chicken died, I'm ashamed to say we cooked her and ate her. I mean it's no more or less respectful than burying her, and we gave her a full life and lots of love! Anyway the point is I was never sure that we cooked my first pet Sally the Chicken at the right temperature, because she tasted strange. I always remember that the temperature was the number on the back of my debit card, 351. That also happens to be the last three digits of my

social security number. Talk about a coincidence! Anywho I've got to go visit my first grade teacher Martha, and then run by the bank."

These utterances rely on the ambiguity inherent in natural language: homophones, homonyms, and conversations without clearly delineated topic boundaries. Though our study tested for relatively simplistic attacks, these utterances could have detected misbehavior of a much more subtle nature, which demonstrates the benefits of a human-in-the-loop approach.

At the conclusion of the study, we asked participants for their overall opinions about their Test Drive experience, expressed in a Likert-scale question and followup free response. The overwhelming majority found the interface to be somewhat or very useful for the purpose of identifying malicious apps (76%, $n = 140$) or in general (87%, $n = 60$). In open-ended responses, participants stated that the interface "*would help [them] decide what apps are malicious*":

- "*I would definitely use it before installing an app. I wouldn't feel comfortable speaking around my Alva [the name of the hypothetical device in our study] otherwise in case I accidentally say something that would leak my information or put me in a bad light.*" (P196)
- "*I would [perform Test Drives]. In fact, I would be even more thorough than in this test as the thought of someone actually using my information without my knowledge terrifies me. As such, I would type in a wide range of sample sentences — both routine and those that would not come up in regular conversation — to test it.* " (P75)

Others felt that Test Drives could be a useful way to supplement other sources of information:

- "*I would use it and then try to confirm online with user reviews.*" (P121)
- "*I wouldn't trust [Test Drives] completely for making my decision about whether or not use the app, but it would be a place to start with getting more information on the app.*" (P156)

However, some respondents emphasized that they felt Test Drives on their own would not provide sufficient information:

- "*I would want each app test driven by thousands of people before installing. It would be easy to dupe an individual or even a few.*" (P181)
- "*No. I'm not able to test it thoroughly enough to get a reliable determination of whether it's malicious or not. I would have to rely on an expert to do proper rigorous testing to know for sure.*" (P73)

Still others said that they distrusted the entire concept of always-listening devices, and Test Drives did not sway their opinion:

- "*I honestly wouldn't use Alva because of my privacy concerns, but if I did I would use this interface to test things I commonly say.*" (P192)
- "*I wouldn't own Alva. I'm highly aware that my personal information is already more readily available in this world of technology and social media than I would like. Adding on Alva seems like a completely unnecessary extra risk. If I did anyways? I suppose I would use Test Drive, because then at least you can filter out the more obvious malicious softwares. But I still wouldn't trust it.*" (P197)

While participants responded positively to the Test Drive interface, and many clearly expended significant effort, did this effort translate into success at detecting malicious apps? We investigate this question in the next section.

5.2 Can people detect malicious AI on their own?

Overall, our study's results suggest that people are able to detect malicious apps, but performance varies significantly and is subject to a variety of nuances.

Table 4. **Detection rates by attack**: percentage of participants in *Test* variant who perceived the attack app as “probably” or “very likely” malicious, broken down by attack condition.

	Detection rate
Financial ($n = 16$)	75%
Sensitive ($n = 18$)	44%
PII ($n = 13$)	46%
Overcapture ($n = 13$)	7.7%
Total ($n = 60$)	45%

Table 5. **Detection rates pre-treatment**: percentage of participants who—on their first try without training—successfully detected a malicious app. Results from collaborative mode and the *Test* variant of individual mode are combined, as their task was identical.

	Detection rate
Financial ($n = 38$)	45%
Sensitive ($n = 34$)	32%
PII ($n = 38$)	16%
Overcapture ($n = 30$)	10%
Total ($n = 140$)	26%

We first consider the overall accuracy of individuals, further focusing on those whose primary task was evaluating apps’ maliciousness (*Test* variant, $n = 60$). Recall that, in the second phase, participants were asked to evaluate two different apps, only one of which was malicious. As seen in Figure 5, almost half (45%) correctly identified the malicious app. Only a small minority believed the benign app to be malicious, for a false positive rate of 6.7%.

Participants had varying success in identifying different types of attacks. Table 4 shows that while only 1 person out of 13 detected the *Overcapture* attack, three quarters of those who experienced the *Financial* attack correctly detected it. (This difference between the four attack conditions is statistically significant, $\chi^2(3) = 13.1, p = 0.00435$.)

By this point in the study, participants had already learned about one type of attack. Because the four attack conditions were assigned randomly *with replacement*, a quarter of participants (in expectation) faced an attack they were familiar with. Figure 6 shows that participants were much more likely (but not guaranteed) to detect an attack to which they had been previously exposed. We verified that this difference is statistically significant using Fisher’s exact test (odds ratio = $0.143, p = 0.00112$).

Many fewer participants were able to detect attacks before the informational treatment (Table 5): on average, their success rate was only 26%, compared with 45% after the treatment. We verified that this difference was statistically significant by comparing participants’ performance before and after treatment using the Wilcoxon signed-rank test ($W = 13.1, p = 0.0231$).

The primary metric used in the analysis so far is participants’ *perception*: whether they thought an app was malicious. However, this metric may be noisy. Sometimes, people perceive an app as

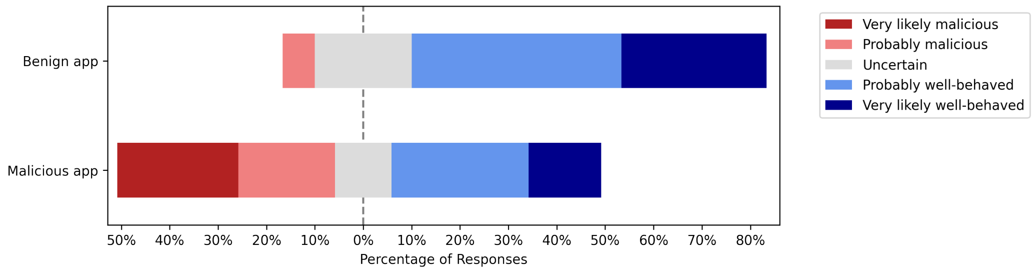


Fig. 5. **Perceptions of maliciousness**: Participants from the *Test* variant ($n = 60$), during the two post-treatment Test Drives, expressing their perceptions of whether the app they tested was malicious.

Table 6. **Install vs test**: percentage of participants who *perceived* the attack app as malicious, separated between the *Install* and *Test* variants, and further broken down by attack condition.

	Install variant ($n = 60$)	*Test* variant ($n = 60$)
Financial	21%	75%
Sensitive	31%	44%
PII	17%	46%
Overcapture	13%	7.7%
Total	20%	45%

misbehaving because it heard something they consider out of scope—but the researcher making the judgment about relevance *did* consider it in scope. Conversely, people may see evidence of misbehavior but not interpret it as an attack. To understand the extent to which these scenarios may have affected our results, we defined and examined a new metric: attack *discovery*. We consider a participant in individual mode to have discovered an attack if they submitted an utterance that matches the malicious app's attack behavior (for example, mentioning bank details when the app is listening for financial information).[8] Using this metric, we found that, compared with the fraction of people who perceived the app to be malicious (45%), an approximately similar proportion of participants *discovered* examples of it misbehaving (50%). (A complete breakdown is available in Tables **??** and **??** in the Supplementary Materials.)

5.3 How does task formulation affect detection?

Our analysis so far has focused on participants whose task was defined as evaluating whether an app was malicious ("Try enough inputs to make up your mind about whether this app is malicious or not."). But for some of our participants ($n = 60$), we defined their primary task as deciding about installing the app ("Try enough inputs for you to make up your mind about whether or not you would install this app."). The detection rate in this variant was less than half of that in the *Test* variant (Table 6), and the majority appeared willing to install the malicious app. We verified that the difference between the two variants is statistically significant using Fisher's exact test (odds ratio = 3.27, $p = 0.00299$). This suggests that people are significantly more effective at detecting misbehaving apps when this is their primary task.

[8]Utterances where "The app would hear *only part* of this" were considered successful discoveries for this metric.

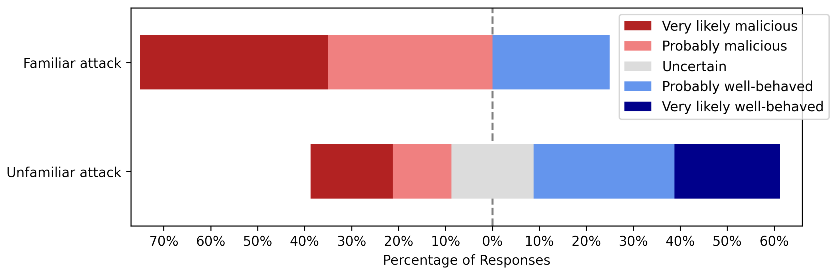


Fig. 6. **Effects of familiarity on detection**: Participants from the *Test* variant ($n = 60$), expressing their perceptions of whether the *malicious* app they tested was malicious—separated between those encountering a familiar attack and those for whom it was novel.

Table 7. **Group detection rates by condition**: collaborative mode participants ($n = 80$) who correctly identified the malicious app.

	Detection rate
Financial ($n = 20$)	90%
Sensitive ($n = 20$)	40%
PII ($n = 20$)	60%
Overcapture ($n = 20$)	10%

The *Install* variant encompassed two different versions: for a subset of participants, we referred to apps as "misbehaving" rather than "malicious." We found that, compared with the "misbehaving" wording ($n = 40$), twice as many participants with the "malicious" wording ($n = 20$) identified the attack app after seeing the explanation (30% vs 15%). However, this difference was not statistically significant (Fisher's exact test, $p = 0.15$).

Within the *Test* variant, we tried different nudges to make participants more effective at identifying malicious apps. Participants (20 per variant) saw either the "control" option (no nudges), a suggestion to think about attacks other than those they had seen previously, or a reminder of what they said about not wanting the device to hear. To examine the effects of these nudges, we performed a logistic regression, with detection as the outcome and nudge variant and attack condition as the predictor variables. The regression ($R^2 = 0.228$) found a weak positive effect from the two nudges, but the effects were not statistically significant ($z = 1.75, p = 0.080$ and $z = 1.619, p = 0.106$). This suggests that neither reminding people about information they consider private, nor encouraging them to be creative with their attacks, makes much of a difference on their ability to detect malicious apps.

5.4 How well does collaborative detection work?

In the second part of our study, participants performed Test Drives collaboratively. We found that, overall, this resulted in higher performance. In collaborative mode, 50% correctly detected the attack app, and the false positive rate (perceiving the benign app as malicious) was 8.8% (see Figure 7). Table 7 shows the detection rates by condition, which are similar in magnitude and distribution to results from individual mode (cf. Table 4).

Collaborative mode offered participants the opportunity to report utterances that provided evidence of malicious behavior (and required them to do so if they rated the app as "probably" or "very likely" malicious). This enables a new discovery metric: we can say that a participant

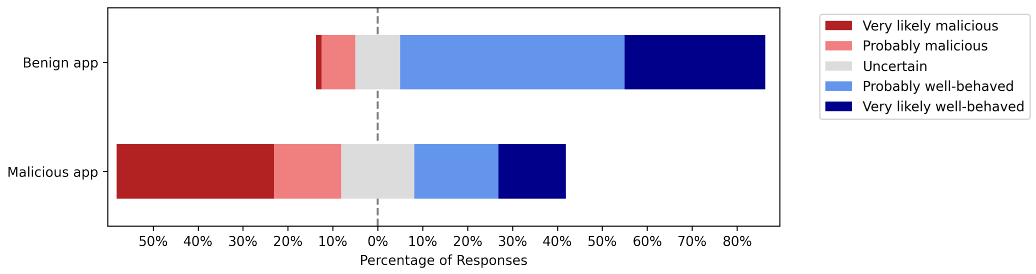


Fig. 7. **Perceptions of maliciousness in collaborative mode**: Participants in collaborative mode ($n = 80$), during the two post-treatment Test Drives, expressing their perceptions of whether the app they tested was malicious.

Table 8. **Minimum group size**: which participant in the group was the first to have noticed the app was malicious? This tells us how small this group could have been and still noticed the attack.

Condition	Group	Min. size
Financial	A	1
	B	3
Sensitive	A	4
	B	1
PII	A	1
	B	4
Overcapture	A	7
	B	4

discovered an attack if they reported a relevant utterance. Using this metric, the percentage of collaborative mode participants who reported a maliciously-heard utterance (i.e., the true positive rate) was 66%. The corresponding false positive rate is the fraction of participants who reported an utterance from the benign app; this value was 18%. Additionally, 18% of participants reported (what we consider to be) a benign utterance from a malicious app. (The latter two groups overlap, but only partially, so that a total of 31% participants made some sort of false report.)

Since the intent of collaborative mode is for evaluators to build on each other's efforts, a natural way of evaluating their performance is to ask whether the group, as a whole, detected an attack. The easiest way to define this is by saying that a group was able to detect an attack if at least one of its members successfully reported it. Using this metric, 100% of our groups (8 out of 8) were able to detect the attack. (As noted in Section 3.6, each of the four attacks was evaluated by two completely independent groups.) If we use this approach, then the corresponding false positive rate can be calculated by counting the number of groups where at least one person falsely reported an utterance from the benign app; that number is 6 out of 8, for a false positive rate of 75%.

There are other metrics for defining a group's success that may be preferable, for example by being less noisy and representing when the group achieves consensus. Some candidates include:

- when a threshold number of group members report the app as malicious
- when a threshold number of group members report a specific utterance as malicious
- when a threshold number of those who have seen a specific utterance report it as malicious

Each group in collaborative mode consisted of 10 participants. We saw above that even groups of this modest size had at least one person successfully detect the attack. However, the group size was fairly arbitrary; what would have happened had we chosen a different cut-off? To answer this, we ordered participants based on the number of utterances they saw. Then, we repeated our analysis by iteratively removing the last participant and seeing whether this smaller subgroup would have detected the attack. Table 8 shows the results of this analysis. In all but one condition, every group detected the attack within the first four participants who tested the app.

Another way of measuring the group's effort is the number of *unique* examples of malicious behavior they were able to uncover. Figure 8 shows this quantity as a function of group size. After the first examples of malicious behavior were identified, groups continued submitting new candidates, though not every member chose to do this.

One more question is whether there was consensus among the group about the malicious utterances: did everyone agree they were examples of misbehavior? To answer this, we define a new metric to capture consensus among reporters: number of utterances reported by over half of those who saw them. Figure 9 shows this data, again as a function of group size. We see that, in some groups, consensus was hard to come by, with low levels of agreement.

Finally, we consider whether people perform better as individuals or as groups. The answer is not trivial because individuals submit more utterances and may be less susceptible to informational cascades, but people in groups may learn from each other. Comparing the probability that a set of people—drawn either from individual or from collaborative mode—are able to detect an attack, we found that, for very small group sizes, people may be more successful working on their own, but, for groups with more than 4 members, the collaborative effort of the group exceeds an individual's performance (see Figure **??** in Supplementary Materials for details).

5.5 How well does detection work when there is no interactive feedback?

Near the end of our study, we asked participants to submit attack utterances for a fourth app.[9] The task (even in the *Install* variant[10]) was to determine whether the app was malicious. But, unlike in the Test Drives, there was no interactive feedback about whether the app would hear the utterances: the participants were just asked to submit utterances they would use for testing.

[9]Its purpose was described as providing film and television recommendations.
[10]We introduced this procedure after collecting the first 20 data points in our study, leaving $n = 180$ for this experiment;

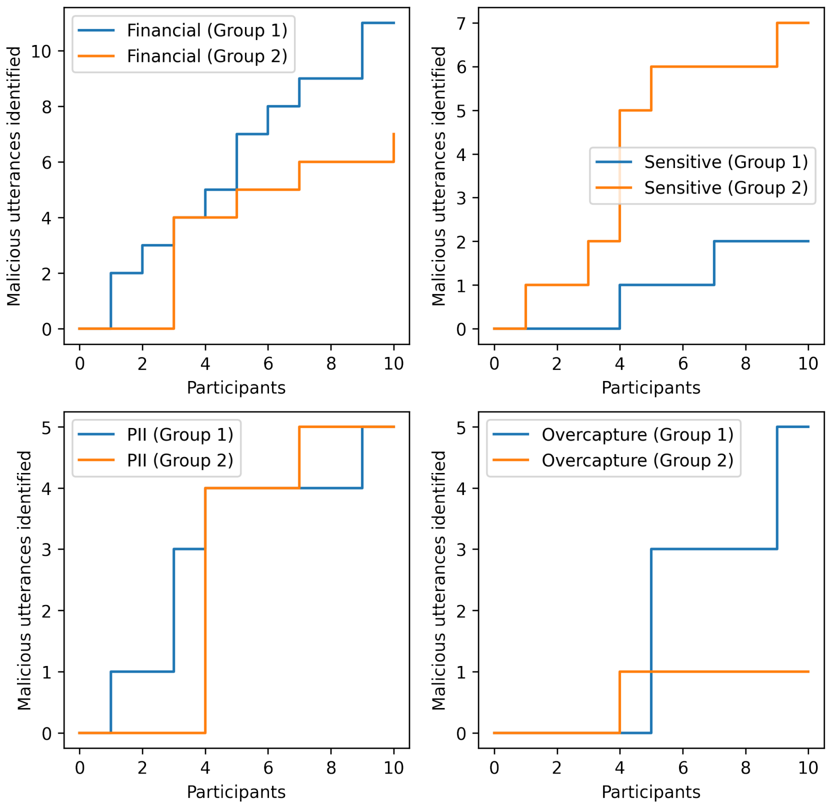


Fig. 8. **How many malicious utterances were identified as group size increased?** This figure shows the total number of malicious utterances that the group identified (i.e., someone wrote an utterance, the Test Drive stated that the app would hear it, and this behavior was reported as inappropriate) as participants were added until the group reached its final size of 10 people.

Table 9. **Detection rates for non-interactive utterances**: percentage of participants (n = 180), who—during the non-interactive stage—submitted an utterance targeting each attack type. This therefore tells us how many participants would have successfully detected the attack, had it been of that particular type.

	Detection rate
Financial	69.4%
Sensitive	52.8%
PII	45.6%
Overcapture	23.3%

This experiment allowed us to see whether participants could generate effective attacks in the absence of interactive feedback. Table 9 shows how many participants would have discovered a malicious application of each of the four attack types. These non-interactive detection rates are broadly similar to performance in interactive Test Drives in the *Test* variant (Table 4).

Since this stage took place at the end of the study, we could also use this opportunity to see whether participants had learned new testing techniques over the course of the study. We can look for evidence of learning by comparing the number of different attacks submitted during the non-interactive stage to the same metric from the very first Test Drive, before all treatments. The mean

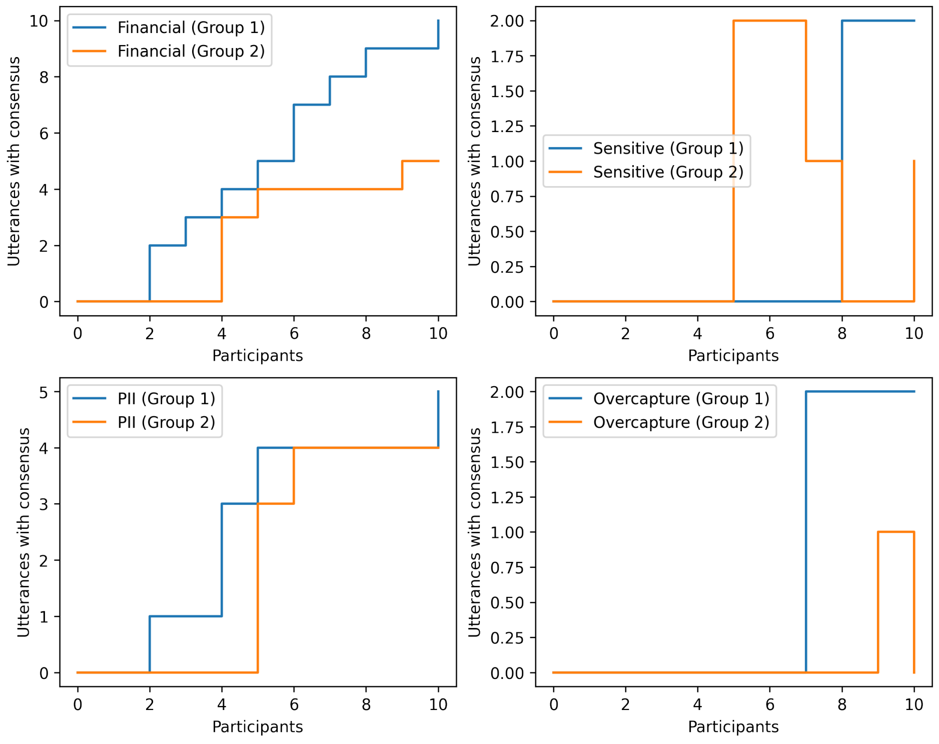


Fig. 9. **Was there consensus about which utterances were malicious?** This figure shows the total number of utterances that were reported as malicious by over half of those who saw them, as a function of group size. A decrease is observed when participants are added to the group but do not report utterances as malicious, therefore diluting the previously existing consensus.

number of attack types submitted during the non-interactive stage was 2.1; pre-treatment, the same participants used, on average, 1.7 attack types. We compared the pairs of samples using a Wilcoxon signed-rank test and found that the difference was significant ($W = 2,954.5, p = 0.000723$).

6 DISCUSSION

The utterances submitted by participants, as well as their answers to open-ended questions in our survey, show that nearly all effectively understood the hypothetical always-listening services involved in our study and the Test Drive task itself. As noted in Section 5.1, just about everyone submitted attacks (92%) and most tried multiple types of attacks (2.6, on average). One implication of this finding is that Test Drives can be performed by people without technical or security backgrounds. For platforms, this means that crowdsourcing the evaluation could be a practical choice. Test Drives therefore join the literature on crowdsourced security evaluations [3, 22, 37] as another promising technique. Our results also suggest that Test Drives could be incorporated into consumer-facing systems and interfaces without causing confusion.

One of the open questions in our study was the target audience for Test Drives: end-users or testers employed by platforms? Our results suggest that both may be appropriate, but in different ways. The best way to uncover malicious apps is for platforms to employ dedicated testers. Because most users expend limited effort on testing, they may discover straightforward attacks while overlooking more complex and nuanced privacy violations. Having large numbers of users can help with this problem but may not be able to solve it entirely. Dedicated testers are able to spend more time and effort on this task, and can benefit from specialized training.

However, end-users can also get value from Test Drives. The vast majority of participants found Test Drives to be a useful mechanism for understanding the behavior of always-listening apps: over three quarters found it somewhat or very useful. Those surveyed were also enthusiastic in their free-response answers. However, while the mechanism received praise, a number of respondents also cautioned that Test Drives should be just one facet of protection among many, and that they would look for other factors, such as reviews, to make their decisions.

Test Drives furthermore provide non-security value to users: many used them to supply positive examples of things the app should have heard. This shows that offering users the opportunity to Test Drive apps before installation can build trust and instill confidence in the system, which would benefit platforms even if end-users uncover few violations on their own. Moreover, this behavior can be harnessed as a useful signal by platforms: if an app fails to find many positive examples relevant, this is an indicator of some defect, even if it may or may not be a security issue. This implies that platforms may wish to support Test Drive behaviors, even if adoption of the exact architecture proposed in this paper may not be practical. For example, Amazon already provides an Alexa Simulator [4] that allows developers to test skills in a sandbox; opening this functionality to consumers could prove to be useful.

Success rates at detecting malicious apps varied across the board, from 7.7% to 75%. The higher detection rates are encouraging and highlight the fact that people pay attention and are able to use the Test Drive interface to detect violations—at least more egregious ones. The lower numbers are obviously less promising, but there are some mitigating factors. First, the lower-performing conditions posed less of a privacy threat. Second, a non-trivial number of participants *were* able to detect these malicious apps, and the collaborative mode experiments showed that their findings can effectively be amplified in group settings.

Our results also demonstrate the importance of training in the detection task. Even our short informational treatment helped improve our participants' success rate from 26% to 45%. This holds the promise that, with extra training and experience, people will improve their performance even further. Such training may include learning about different types of attacks and the information attackers may target, and perusing a variety of test cases. Existing literature and curricula on teaching privacy may offer relevant materials and approaches [19, 69]. In particular, participants in our study did much better at discovering attack types that they had encountered before. Therefore, in a real app store, as soon as a malicious app is discovered, its behavior and relevance detector should be shared with testers, so they can learn from it. We also note that how the task is specified appears to influence outcomes (Section 5.3), which suggests that materials need to be carefully designed and workshopped. Overall, this is another reason why app evaluation should be done by platforms: they have the capacity and budgets to train dedicated workers to detect malware.

People's success rate at detecting attacks varied by the specific attack type. Most people detected financial attacks easily and found other attacks more difficult; the "overcapture" attack was especially challenging. Users' prior knowledge offers one explanation, since many are familiar, from media reports, of hackers trying to steal financial information. Another possible explanation is that some harms are more visceral than others. For example, having one's financial details breached could be readily interpreted as harmful, compared to the subtleties of having their personally identifiable information revealed publicly. Since such harms come to mind less readily, people may therefore be less likely to search for examples of them—a manifestation of the availability heuristic.

One reason that so few people detected "overcapture" attacks was that they required multiple topics to "trigger" the attack. The chance of an utterance including multiple topics is higher in longer and more complicated utterances, but most inputs in our study were shorter—single sentences or even phrases—demonstrating a clear bias. This suggests that, if we want testers to generate longer and more complex test cases, they will need additional training or some form of assistance from the Test Drive interface. Another solution is to help testers synthesize longer inputs, for example by using their own conversations that their device has previously recorded.

Our study found that people working in groups had modestly higher success rates than those working alone (50% and 45%, respectively). More notably, groups were able to achieve these detection rates with dramatically less effort on the part of each individual participant: six utterances fewer per person, on average. Moreover, working in groups gave users the opportunity to learn from others. In the words of one participant, "*After my personal Test Drive, I understood what phrases were better for testing if an app was malicious or not based on others' inputs.*" The newly acquired techniques may then prove to be useful in future Test Drives.

Any detection process has the potential for false positives; Test Drives are also vulnerable to this problem, with observed false positive rates of 6.7% and 8.8% for individuals and groups, respectively. (The latter rate is higher for individual utterances, at 31%, suggesting that app reports are more reliable.) A potential explanation is that people may be flagging examples that have more sensitive information but are relevant to the app. While it remains an open question how these rates might compare to more automated detection methods, any deployed system will need to be prepared for this. Potential defenses include having a threshold for reports, requiring some sort of consensus, or having multiple tiers of reviews.

7 CONCLUSION AND FUTURE WORK

Intelligent voice assistants are constantly adding new features and are progressing towards adopting passive-listening capabilities, in which ambient conversations are monitored and analyzed. They also have well-developed ecosystems of third-party applications, and it is plausible that platforms will wish to extend some always-listening capabilities to them. The goal of this study was to explore how this can be done safely, without enabling potentially malicious apps to spy on their users.

We argued that a comprehensive model of privacy requires apps to hear only things that are relevant to them—nothing more. However, making the determination of what is relevant to any given app remains a difficult problem for NLP algorithms; for now, it is most suitable to human judgment. But the question of how well humans can exercise that judgment, and whether they can uncover violations, has not previously been explored. We therefore set out to test it.

We first proposed a system architecture in which apps were split into relevance detectors and feature modules. This provided a guarantee that any classification would be idempotent, reproducible, and not subject to subterfuge on the part of the app.

Next, we introduced an interface, the Test Drive, through which people can test the behavior of an app's relevance detector. We simulated this interface using the Wizard of Oz technique, in order to study how people would use it and whether they could detect several basic types of attacks.

We found that people mostly used the Test Drive interface to examine whether the app worked as advertised, unless they were explicitly told to see if it was malicious. Most commonly, the inappropriate behavior people looked for involved financial details, resulting in high success rates at identifying misbehaving apps that targeted this information. People were moderately successful at finding other attacks, such as those targeting PII and sensitive conversation topics. Subtle attacks had a low success rate and may be less well-suited to Test Drives.

Our study also found that people were more effective at discovering malicious apps when they could build on the experiences of others, leading us to conclude that evaluations should be done collaboratively, ideally by trained workers employed by platforms. However, participants responded positively to Test Drives, and we found that they enabled user trust. We therefore believe that they would be welcomed by end-users as a way to try out apps before installing them.

Our results raise a number of open questions that future work can explore.

We found that users view Test Drives positively and could see themselves utilizing this interface before installing passive-listening applications. While such apps are still hypothetical, existing voice assistants feature tens of thousands of apps, and users are deciding daily about whether to adopt them. How can Test Drives be incorporated into present systems and interfaces? For example, existing Alexa skills can be run in a simulator [56], but this is largely targeted at developers. What is the best way to offer this capability to users, and how would they take advantage of it?

We concluded that, for the purpose of identifying malicious apps, collaborative evaluation efforts are likely to be more successful. How should these collaborative Test Drives be organized? In particular, how many people need to be involved? Is it better to coordinate their efforts or let them proceed organically? Is it preferable for them to work in parallel or in sequence? What kind of training is most effective for helping detect malicious apps?

While we have argued that human judgment is necessary for making accurate relevance determinations, we also believe that the process for identifying malicious apps need not rely on humans alone: it can be much less manual than simply offering the Test Drive interface. In a complete system, human judgment would guide, or be supplemented by, additional algorithmic testing. Which parts of the process can be automated and which require human input? Perhaps people will create the base examples, and algorithms will permute and rearrange them, to create a variety of similar inputs on their basis (to ensure test cases are not word-choice dependent). Or people will define conversational contexts, and NLP algorithms will be able to write entire conversations within them. Potentially, humans may be brought in only to test subtle edge cases, while the bulk of testing will rely on pre-written and automated examples. What is the best way for human and algorithm to complement each other?

This study has focused on always-listening services, but they are just one area where malicious algorithms, including those powered by artificial intelligence, pose privacy and security issues. How can Test Drives be adopted to other AI domains? We believe that our observations are likely to be applicable to other areas that rely on machine learning and which offer opportunities to decompose larger problems into smaller, self-contained tasks that are amenable to human verification. Algorithmic decision-making, machine translation, detection of toxic comments, household robots, and even self-driving cars are all examples where people have to trust black-box algorithms. In all of these cases, offering a Test Drive option can help win users' trust. Each of these domains provides the possibility to define an isolated test instance, examine the model's behavior under these circumstances, and subject its choices to human scrutiny to see if it is behaving in a potentially malicious manner. A key requirement is that the functionality tested is stateless, so that it cannot game the system by altering its behavior based on time or usage level. In fact, this notion of statelessness and separability may itself be a lesson for the design of Artificial Intelligence: to make an AI that is understandable, trustworthy, and can be shown to not be malicious, design it in a way that allows users to take it for a Test Drive.

ACKNOWLEDGMENTS

We are grateful to Conor Gilsenan for edits to a draft of this paper and to Gary Liu for assistance with data analysis. This work was supported by NSF grant CNS-1801501, NSA contract H98230-18-D-0006, the Center for Long-Term Cybersecurity at UC Berkeley, and generous gifts from Cisco and Mozilla.

REFERENCES

[1] Noura Abdi, Kopo M. Ramokapane, and Jose M. Such. 2019. More than Smart Speakers: Security and Privacy Perceptions of Smart Home Personal Assistants. In *Fifteenth Symposium on Usable Privacy and Security (SOUPS 2019)*. USENIX Association. https://www.usenix.org/conference/soups2019/presentation/abdi

[2] Rebecca Adaimi, Howard Yong, and Edison Thomaz. 2021. Ok Google, What Am I Doing? Acoustic Activity Recognition Bounded by Conversational Assistant Interactions. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 5, 1, Article 2 (March 2021). https://doi.org/10.1145/3448090

[3] Yuvraj Agarwal and Malcolm Hall. 2013. ProtectMyPrivacy: Detecting and Mitigating Privacy Leaks on IOS Devices Using Crowdsourcing. In *Proceeding of the 11th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '13)*. Association for Computing Machinery, New York, NY, USA, 97–110. https://doi.org/10.1145/2462456.2464460

[4] Amazon. [n. d.]. Alexa Simulator. https://developer.amazon.com/en-US/docs/alexa/devconsole/alexa-simulator.html

[5] Tawfiq Ammari, Jofish Kaye, Janice Y. Tsai, and Frank Bentley. 2019. Music, Search, and IoT: How People (Really) Use Voice Assistants. *ACM Transactions on Computer-Human Interaction* 26, 3, Article 17 (April 2019), 28 pages. https://doi.org/10.1145/3311956

[6] Salvatore Andolina, Valeria Orso, Hendrik Schneider, Khalil Klouche, Tuukka Ruotsalo, Luciano Gamberini, and Giulio Jacucci. 2018. Investigating Proactive Search Support in Conversations. In *Proceedings of the 2018 Designing Interactive Systems Conference (DIS '18)*. ACM, 1295–1307. https://doi.org/10.1145/3196709.3196734

[7] Samuel Axon. 2020. Amazon Halo Will Charge a Subscription Fee to Monitor the Tone of Your Voice. *Ars Technica* (Aug. 2020). https://arstechnica.com/gadgets/2020/08/amazon-halo-will-charge-a-subscription-fee-to-monitor-the-tone-of-your-voice/

[8] Sushil Bikhchandani, David Hirshleifer, and Ivo Welch. 1998. Learning from the Behavior of Others: Conformity, Fads, and Informational Cascades. *Journal of Economic Perspectives* 12, 3 (Sept. 1998), 151–170. https://doi.org/10.1257/jep.12.3.151

[9] Barry Brown, Moira McGregor, and Donald McMillan. 2015. Searchable Objects: Search in Everyday Conversation. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW '15)*. Association for Computing Machinery, New York, NY, USA, 508–517. https://doi.org/10.1145/2675133.2675206

[10] Juan Pablo Carrascal, Rodrigo De Oliveira, and Mauro Cherubini. 2015. To Call or to Recall? That's the Research Question. *ACM Transactions on Computer-Human Interaction* 22, 1, Article 4 (March 2015), 30 pages. https://doi.org/10.1145/2656211

[11] Varun Chandrasekaran, Suman Banerjee, Bilge Mutlu, and Kassem Fawaz. 2021. PowerCut and Obfuscator: An Exploration of the Design Space for Privacy-Preserving Interventions for Smart Speakers. In *Seventeenth Symposium on Usable Privacy and Security (SOUPS 2021)*. USENIX Association, 535–552. https://www.usenix.org/conference/soups2021/presentation/chandrasekaran

[12] Yuxin Chen, Huiying Li, Shan-Yuan Teng, Steven Nagels, Zhijing Li, Pedro Lopes, Ben Y. Zhao, and Haitao Zheng. 2020. Wearable Microphone Jamming. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3313831.3376304

[13] Long Cheng, Christin Wilson, Song Liao, Jeffrey Young, Daniel Dong, and Hongxin Hu. 2020. Dangerous Skills Got Certified: Measuring the Trustworthiness of Skill Certification in Voice Personal Assistant Platforms. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. ACM, Virtual Event USA, 1699–1716. https://doi.org/10.1145/3372297.3423339

[14] Catalin Cimpanu. 2019. Alexa and Google Home Devices Leveraged to Phish and Eavesdrop on Users, Again. *ZDNet* (Oct. 2019). https://www.zdnet.com/article/alexa-and-google-home-devices-leveraged-to-phish-and-eavesdrop-on-users-again/

[15] Catalin Cimpanu. 2020. Academics Smuggle 234 Policy-Violating Skills on the Alexa Skills Store. *ZDNet* (July 2020). https://www.zdnet.com/article/academics-smuggle-234-policy-violating-skills-on-the-alexa-skills-store/

[16] Nils Dahlbäck, Arne Jönsson, and Lars Ahrenberg. 1993. Wizard of Oz Studies: Why and How. In *Proceedings of the 1st International Conference on Intelligent User Interfaces (IUI '93)*. Association for Computing Machinery, New York, NY, USA, 193–200. https://doi.org/10.1145/169891.169968

[17] Peter Dockrill. 2019. Newly Released Amazon Patent Shows Just How Much Creepier Alexa Can Get. *Science Alert* (May 2019). https://www.sciencealert.com/creepy-new-amazon-patent-would-mean-alexa-records-everything-you-say-from-now-on

[18] Daniel J. Dubois, Roman Kolcun, Anna Maria Mandalari, Muhammad Talha Paracha, David Choffnes, and Hamed Haddadi. 2020. When Speakers Are All Ears: Characterizing Misactivations of IoT Smart Speakers. *Proceedings on Privacy Enhancing Technologies* 2020, 4 (Oct. 2020), 255–276. https://doi.org/10.2478/popets-2020-0072

[19] Serge Egelman, Julia Bernd, Gerald Friedland, and Dan Garcia. 2016. The Teaching Privacy Curriculum. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education (SIGCSE '16)*. Association for Computing Machinery, New York, NY, USA, 591–596. https://doi.org/10.1145/2839509.2844619

[20] Malin Eiband, Sarah Theres Völkel, Daniel Buschek, Sophia Cook, and Heinrich Hussmann. 2019. When People and Algorithms Meet: User-reported Problems in Intelligent Everyday Applications. In *Proceedings of the 24th International Conference on Intelligent User Interfaces (IUI '19)*. Association for Computing Machinery, New York, NY, USA, 96–106. https://doi.org/10.1145/3301275.3302262

[21] Sascha Fahl, Marian Harbach, Yasemin Acar, and Matthew Smith. 2013. On the Ecological Validity of a Password Study. In *Proceedings of the Ninth Symposium on Usable Privacy and Security (SOUPS '13)*. ACM, Newcastle, United Kingdom, 13:1–13:13. https://doi.org/10.1145/2501604.2501617

[22] Matthew Finifter, Devdatta Akhawe, and David Wagner. 2013. An Empirical Study of Vulnerability Rewards Programs. In *22nd USENIX Security Symposium (USENIX Security 13)*. USENIX Association, Washington, D.C., 273–288. https://www.usenix.org/conference/usenixsecurity13/technical-sessions/presentation/finifter

[23] Matthew Groh, Ziv Epstein, Chaz Firestone, and Rosalind Picard. 2022. Deepfake Detection by Human Crowds, Machines, and Machine-Informed Crowds. *Proceedings of the National Academy of Sciences* 119, 1 (Jan. 2022), e2110013119. https://doi.org/10.1073/pnas.2110013119

[24] Zhixiu Guo, Zijin Lin, Pan Li, and Kai Chen. 2020. SkillExplorer: Understanding the Behavior of Skills in Large Scale. In *29th USENIX Security Symposium (USENIX Security 20)*. USENIX Association, 2649–2666. https://www.usenix.org/conference/usenixsecurity20/presentation/guo

[25] Majid Hatamian, Jetzabel Serna, and Kai Rannenberg. 2019. Revealing the Unrevealed: Mining Smartphone Users Privacy Perception on App Markets. *Computers & Security* 83 (2019), 332–353. https://doi.org/10.1016/j.cose.2019.02.010

[26] Yue Huang, Borke Obada-Obieh, and Konstantin (Kosta) Beznosov. 2020. Amazon vs. My Brother: How Users of Shared Smart Speakers Perceive and Cope with Privacy Risks. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376529

[27] J. F. Kelley. 1983. An Empirical Methodology for Writing User-Friendly Natural Language Computer Applications. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '83)*. Association for Computing Machinery, New York, NY, USA, 193–196. https://doi.org/10.1145/800045.801609

[28] Sean Michael Kerner. 2018. Researchers Find Amazon Alexa Can Be Hacked to Record Users. *eWeek* (April 2018). https://www.eweek.com/security/researchers-find-amazon-alexa-can-be-hacked-to-record-users

[29] Jonathan Kilgour, Jean Carletta, and Steve Renals. 2010. The Ambient Spotlight: Queryless Desktop Search from Meeting Speech. In *Proceedings of the 2010 International Workshop on Searching Spontaneous Conversational Speech (SSCS '10)*. Association for Computing Machinery, New York, NY, USA, 49–52. https://doi.org/10.1145/1878101.1878112

[30] Saranga Komanduri, Richard Shay, Patrick Gage Kelley, Michelle L. Mazurek, Lujo Bauer, Nicolas Christin, Lorrie Faith Cranor, and Serge Egelman. 2011. Of Passwords and People: Measuring the Effect of Password-Composition Policies. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, Vancouver, BC, Canada, 2595–2604. https://doi.org/10.1145/1978942.1979321

[31] Deguang Kong, Lei Cen, and Hongxia Jin. 2015. AUTOREB: Automatically Understanding the Review-to-Behavior Fidelity in Android Applications. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications*

Security (CCS '15). Association for Computing Machinery, New York, NY, USA, 530–541. https://doi.org/10.1145/2810103.2813689

[32] Deepak Kumar, Riccardo Paccagnella, Paul Murley, Eric Hennenfent, Joshua Mason, Adam Bates, and Michael Bailey. 2018. Skill Squatting Attacks on Amazon Alexa. In *27th USENIX Security Symposium (USENIX Security 18)*. USENIX Association, 33–47. https://www.usenix.org/conference/usenixsecurity18/presentation/kumar

[33] Josephine Lau, Benjamin Zimmerman, and Florian Schaub. 2018. Alexa, Are You Listening?: Privacy Perceptions, Concerns and Privacy-seeking Behaviors with Smart Speakers. *Proc. ACM Hum.-Comput. Interact.* 2, CSCW (Nov. 2018), 102:1–102:31. https://doi.org/10.1145/3274371

[34] Lily Hay Newman. 2018. Turning an Echo Into a Spy Device Only Took Some Clever Coding. *Wired* (April 2018). https://www.wired.com/story/amazon-echo-alexa-skill-spying/

[35] Lily Hay Newman. 2020. An Alexa Bug Could Have Exposed Your Voice History to Hackers. *Wired* (Aug. 2020). https://www.wired.com/story/amazon-alexa-bug-exposed-voice-history-hackers/

[36] Yuchen Liu, Ziyu Xiang, Eun Ji Seong, Apu Kapadia, and Donald S. Williamson. 2021. Defending Against Microphone-Based Attacks with Personalized Noise. *Proceedings on Privacy Enhancing Technologies* 2021, 2 (April 2021), 130–150. https://doi.org/10.2478/popets-2021-0021

[37] Thomas Maillart, Mingyi Zhao, Jens Grossklags, and John Chuang. 2017. Given Enough Eyeballs, All Bugs Are Shallow? Revisiting Eric Raymond with Bug Bounty Programs. *Journal of Cybersecurity* 3, 2 (Oct. 2017), 81–90. https://doi.org/10.1093/cybsec/tyx008

[38] David J. Major, Danny Yuxing Huang, Marshini Chetty, and Nick Feamster. 2019. Alexa, Who Am I Speaking To? Understanding Users' Ability to Identify Third-Party Apps on Amazon Alexa. (Oct. 2019). http://arxiv.org/abs/1910.14112

[39] Nathan Malkin, Joe Deatrick, Allen Tong, Primal Wijesekera, Serge Egelman, and David Wagner. 2019. Privacy Attitudes of Smart Speaker Users. *Proceedings on Privacy Enhancing Technologies* 2019, 4 (2019), 250–271. https://doi.org/10.2478/popets-2019-0068

[40] Nathan Malkin, Serge Egelman, and David Wagner. 2019. Privacy Controls for Always-Listening Devices. In *Proceedings of the New Security Paradigms Workshop (NSPW '19)*. Association for Computing Machinery, New York, NY, USA, 78–91. https://doi.org/10.1145/3368860.3368867

[41] David Maulsby, Saul Greenberg, and Richard Mander. 1993. Prototyping an Intelligent Agent through Wizard of Oz. In *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems (CHI '93)*. Association for Computing Machinery, New York, NY, USA, 277–284. https://doi.org/10.1145/169059.169215

[42] Moira McGregor and John C. Tang. 2017. More to Meetings: Challenges in Using Speech-Based Technology to Support Meetings. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing (CSCW '17)*. ACM, New York, NY, USA, 2208–2220. https://doi.org/10.1145/2998181.2998335

[43] Donald McMillan, Antoine Loriette, and Barry Brown. 2015. Repurposing Conversation: Experiments with the Continuous Speech Stream. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 3953–3962. https://doi.org/10.1145/2702123.2702532

[44] Abraham Mhaidli, Manikandan Kandadai Venkatesh, Yixin Zou, and Florian Schaub. 2020. Listen Only When Spoken To: Interpersonal Communication Cues as Smart Speaker Privacy Controls. *Proceedings on Privacy Enhancing Technologies* (2020), 20.

[45] Richard Mitev, Markus Miettinen, and Ahmad-Reza Sadeghi. 2019. Alexa Lied to Me: Skill-based Man-in-the-Middle Attacks on Virtual Assistants. In *Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security (Asia CCS '19)*. Association for Computing Machinery, New York, NY, USA, 465–478. https://doi.org/10.1145/3321705.3329842

[46] Jared Newman. 2020. You Can Now Buy an Amazon Echo Add-on That Stops Alexa from Listening. *Fast Company* (July 2020). https://www.fastcompany.com/90532150/you-can-now-buy-an-amazon-echo-add-on-that-stops-alexa-from-listening

[47] D. C. Nguyen, E. Derr, M. Backes, and S. Bugiel. 2019. Short Text, Large Effect: Measuring the Impact of User Reviews on Android App Security & Privacy. In *2019 IEEE Symposium on Security and Privacy (SP)*. 555–569. https://doi.org/10.1109/SP.2019.00012

[48] Elleen Pan, Jingjing Ren, Martina Lindorfer, Christo Wilson, and David Choffnes. 2018. Panoptispy: Characterizing Audio and Video Exfiltration from Android Applications. *Proceedings on Privacy Enhancing Technologies* 2018, 4 (Oct. 2018), 33–50. https://doi.org/10.1515/popets-2018-0030

[49] Mike Paxton. 2021. Alexa, Tell Me about the Smart Speaker Market in 2021. https://www.spglobal.com/marketintelligence/en/news-insights/blog/alexa-tell-me-about-the-smart-speaker-market-in-2021

[50] Sarah Perez. 2020. COVID-19 Quarantine Boosts Smart Speaker Usage among U.S. Adults, Particularly Younger Users. *Techcrunch* (April 2020). https://techcrunch.com/2020/04/30/covid-19-quarantine-boosts-smart-speaker-usage-among-u-s-adults-particularly-younger-users/

[51] Jon Porter. 2019. Security Researchers Expose New Alexa and Google Home Vulnerability. *The Verge* (Oct. 2019). https://www.theverge.com/2019/10/21/20924886/alexa-google-home-security-vulnerability-srlabs-phishing-eavesdropping

[52] Irwin Reyes, Primal Wijesekera, Joel Reardon, Amit Elazari Bar On, Abbas Razaghpanah, Narseo Vallina-Rodriguez, and Serge Egelman. 2018. "Won't Somebody Think of the Children?" Examining COPPA Compliance at Scale. *Proceedings on Privacy Enhancing Technologies* 2018, 3 (June 2018), 63–83. https://doi.org/10.1515/popets-2018-0021

[53] Laurel D. Riek. 2012. Wizard of Oz Studies in HRI: A Systematic Review and New Reporting Guidelines. *J. Hum.-Robot Interact.* 1, 1 (July 2012), 119–136. https://doi.org/10.5898/JHRI.1.1.Riek

[54] Sherif Saad, William Briguglio, and Haytham Elmiligi. 2019. The Curious Case of Machine Learning In Malware Detection. (May 2019). http://arxiv.org/abs/1905.07573

[55] Lea Schönherr, Maximilian Golla, Thorsten Eisenhofer, Jan Wiele, Dorothea Kolossa, and Thorsten Holz. 2020. Unacceptable, Where Is My Privacy? Exploring Accidental Triggers of Smart Speakers. *arXiv:2008.00508 [cs]* (Aug. 2020). arXiv:2008.00508 [cs] http://arxiv.org/abs/2008.00508

[56] Faysal Hossain Shezan, Hang Hu, Jiamin Wang, Gang Wang, and Yuan Tian. 2020. Read between the Lines: An Empirical Measurement of Sensitive Applications of Voice Personal Assistant Systems. In *Proceedings of the Web Conference 2020 (WWW '20)*. Association for Computing Machinery, New York, NY, USA, 1006–1017. https://doi.org/10.1145/3366423.3380179

[57] Yang Shi, Yang Wang, Ye Qi, John Chen, Xiaoyao Xu, and Kwan-Liu Ma. 2017. IdeaWall: Improving Creative Collaboration through Combinatorial Visual Stimuli. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing (CSCW '17)*. Association for Computing Machinery, New York, NY, USA, 594–603. https://doi.org/10.1145/2998181.2998208

[58] John M. Simpson. 2017. Home Assistant Adopter Beware: Google, Amazon Digital Assistant Patents Reveal Plans for Mass Snooping. (2017). https://www.consumerwatchdog.org/privacy-technology/home-assistant-adopter-beware-google-amazon-digital-assistant-patents-reveal

[59] Madiha Tabassum, Tomasz Kosiński, Alisa Frik, Nathan Malkin, Primal Wijesekera, Serge Egelman, and Heather Richter Lipford. 2019. Investigating Users' Preferences and Expectations for Always-Listening Voice Assistants. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 4, Article 153 (Dec. 2019), 23 pages. https://doi.org/10.1145/3369807

[60] Chuanqi Tao, Hongjing Guo, and Zhiqiu Huang. 2020. Identifying Security Issues for Mobile Applications Based on User Review Summarization. *Information and Software Technology* 122 (2020), 106290. https://doi.org/10.1016/j.infsof.2020.106290

[61] Christian Tiefenau, Maximilian Häring, Eva Gerlitz, and Emanuel von Zezschwitz. 2019. Making Privacy Graspable: Can We Nudge Users to Use Privacy Enhancing Techniques? *arXiv:1911.07701 [cs]* (Nov. 2019). arXiv:1911.07701 [cs] http://arxiv.org/abs/1911.07701

[62] Tavish Vaidya, Yuankai Zhang, Micah Sherr, and Clay Shields. 2015. Cocaine Noodles: Exploiting the Gap between Human and Machine Speech Recognition. In *9th USENIX Workshop on Offensive Technologies (WOOT 15)*. USENIX Association, Washington, D.C. https://www.usenix.org/conference/woot15/workshop-program/presentation/vaidya

[63] Sarah Theres Völkel, Daniel Buschek, Malin Eiband, Benjamin R. Cowan, and Heinrich Hussmann. 2021. Eliciting and Analysing Users' Envisioned Dialogues with Perfect Voice Assistants. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 254. https://doi.org/10.1145/3411764.3445536

[64] Sarah Theres Völkel, Renate Haeuslschmid, Anna Werner, Heinrich Hussmann, and Andreas Butz. 2020. How to Trick AI: Users' Strategies for Protecting Themselves from Automatic Personality Assessment. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–15. https://doi.org/10.1145/3313831.3376877

[65] T. Walshe and A. Simpson. 2020. An Empirical Study of Bug Bounty Programs. In *2020 IEEE 2nd International Workshop on Intelligent Bug Fixing (IBF)*. 35–44. https://doi.org/10.1109/IBF50092.2020.9034828

[66] R. Wang, Z. Wang, B. Tang, L. Zhao, and L. Wang. 2019. SmartPI: Understanding Permission Implications of Android Apps from User Reviews. *IEEE Transactions on Mobile Computing* (2019), 1–1. https://doi.org/10.1109/TMC.2019.2934441

[67] Jing Wei, Tilman Dingler, Enying Gong, Brian Oldenburg, and Vassilis Kostakos. 2020. Proactive Smart Speakers for Chronic Disease Management: Challenges and Opportunities. *"Mapping Grand Challenges for the Conversational User Interface Community" workshop, CHI 2020* (2020). http://www.speechinteraction.org/CHI2020/papers/Proactive%20Smart%20Speakers%20for%20Chronic%20Disease%20Management(2).pdf

[68] Jing Wei, Tilman Dingler, and Vassilis Kostakos. 2021. Developing the Proactive Speaker Prototype Based on Google Home. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems (CHI EA '21)*. Association for Computing Machinery, New York, NY, USA, Article 292. https://doi.org/10.1145/3411763.3451642

[69] Leah Zhang-Kennedy and Sonia Chiasson. 2021. A Systematic Review of Multimedia Tools for Cybersecurity Awareness and Education. *Comput. Surveys* 54, 1, Article 12 (Jan. 2021). https://doi.org/10.1145/3427920

[70] Marrian Zhou. 2018. Amazon's Alexa Guard Can Alert You If an Echo Detects Smoke Alarm, Breaking Glass. *CNET News* (Dec. 2018). https://www.cnet.com/news/amazons-alexa-guard-can-alert-you-if-an-echo-detects-smoke-alarm-breaking-glass/

A APP DESCRIPTIONS

Table 10 below lists the apps used in our study, with the complete descriptions and examples shown to participants.

Table 10. **Apps** shown during the Test Drives to participants.

	Name	Description	Example
Reminders	Automatic Reminders	The purpose of this app is to automatically add to your calendar any appointments or reminders you mention out loud.	If you say "okay, it's settled, we'll meet next Thursday at noon," the app will add this meeting to your calendar.
Cooking	Chef of the Future	The purpose of this app is to advise you on any questions that come up in the kitchen.	You can ask "Chef" about what goes into recipes, which ingredients you can substitute for others, or for other advice about cooking. If it hears your question ("oh no, I think I added a tablespoon of salt instead of a teaspoon!") it'll remember what you were cooking and advise you accordingly ("don't worry! just add one more cup of water").
Weather	Ambient Weather	The purpose of this app is to keep your phone's weather app updated with any destinations you mention in conversation.	If you're discussing your upcoming ski trip, the app will ensure that your phone's weather widget will show that location. You can also ask it questions directly ("what's the weather in Tahoe?").
Movie rec's	What should I watch next?	This app keeps track of the movies/TV shows/videos you watch, and the opinions you expressed about them. Then when you ask it, "what should I watch next?", it can provide a recommendation for you.	"Hey, did you hear that Parasite won the Oscars this year?" "I didn't, but that's great, I loved that movie!" "Same here, I was so excited when it won!" (If the app heard this conversation, it would recommend films similar to the one mentioned.)

Received January 2022; revised April 2022; accepted August 2022