SkillFence: A Systems Approach to Practically Mitigating Voice-Based Confusion Attacks

ASHISH HOODA, University of Wisconsin–Madison, USA
MATTHEW WALLACE, University of Wisconsin–Madison, USA
KUSHAL JHUNJHUNWALLA, University of Washington, USA
EARLENCE FERNANDES, University of Wisconsin–Madison, USA
KASSEM FAWAZ, University of Wisconsin–Madison, USA

Voice assistants are deployed widely and provide useful functionality. However, recent work has shown that commercial systems like Amazon Alexa and Google Home are vulnerable to voice-based confusion attacks that exploit design issues. We propose a systems-oriented defense against this class of attacks and demonstrate its functionality for Amazon Alexa. We ensure that only the skills a user intends execute in response to voice commands. Our key insight is that we can interpret a user's intentions by analyzing their activity on counterpart systems of the web and smartphones. For example, the Lyft ride-sharing Alexa skill has an Android app and a website. Our work shows how information from counterpart apps can help reduce dis-ambiguities in the skill invocation process. We build SkillFence, a browser extension that existing voice assistant users can install to ensure that only legitimate skills run in response to their commands. Using real user data from MTurk (N = 116) and experimental trials involving synthetic and organic speech, we show that SkillFence provides a balance between usability and security by securing 90.83% of skills that a user will need with a False acceptance rate of 19.83%.

CCS Concepts: • **Security and privacy → Authentication**; **Intrusion/anomaly detection and malware mitigation**.

Additional Key Words and Phrases: Alexa, Skill, Skill-Squatting, Voice Attacks, Defense

ACM Reference Format:
Ashish Hooda, Matthew Wallace, Kushal Jhunjhunwalla, Earlence Fernandes, and Kassem Fawaz. 2022. SkillFence: A Systems Approach to Practically Mitigating Voice-Based Confusion Attacks. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 6, 1, Article 16 (March 2022), 26 pages. https://doi.org/10.1145/3517232

1 INTRODUCTION

Rising in popularity, voice assistants, like Amazon Alexa and Google Home, help users accomplish a variety of tasks using speech as input. Through integrating third-party applications, called skills in Alexa terminology, voice assistants empower their users to access personal information, control physical devices in homes [23], and perform financial transactions [8]. Although these devices are useful, they carry significant security and privacy risks to their users.

Authors' addresses: Ashish Hooda, University of Wisconsin–Madison, 1210 West Dayton St, Madison, Wisconsin, USA, ahooda@wisc.edu; Matthew Wallace, University of Wisconsin–Madison, 1415 Engineering Dr, Madison, Wisconsin, USA, mhwallace@wisc.edu; Kushal Jhunjhunwalla, University of Washington, 185 E Stevens Way NE, Seattle, Washington, USA, kushaljh@cs.washington.edu; Earlence Fernandes, University of Wisconsin–Madison, 1210 West Dayton St, Madison, Wisconsin, USA, earlence@cs.wisc.edu; Kassem Fawaz, University of Wisconsin–Madison, 1415 Engineering Dr, Madison, Wisconsin, USA, kfawaz@wisc.edu.

2474-9567/2022/3-ART16 $15.00
https://doi.org/10.1145/3517232

Beyond traditional computer security vulnerabilities, recent work has shown that voice assistants are vulnerable to *voice-based confusion attacks* [10, 28, 29]. To improve their usability, Voice Assistants auto-enable skills to directly execute after the user speaks [17]. Because of imperfections in the speech interpretation pipeline, the Voice Assistant might execute a skill that the user did not intend or expect. Consider a skill that allows users to interact with the Fitbit account. A malicious skill might have the name *Phitbit*. The speech recognition systems can confuse both skills, and invoke the malicious skill, instead of the true fitness tracking Fitbit skill, without the user noticing. Once activated, such a skill can steal private information [20] or perform other dangerous actions. Similar attacks mimic the voice interface of legitimate skills [28] or rely on semantic interpretation errors of natural language understanding [29].

We identify several fundamental deficiencies in the design of systems like Alexa that contribute to these attacks. First, natural language is ambiguous and speech recognition is prone to errors. Second, there is loose vetting of skill metadata allowing attackers to impersonate legitimate skills. Third, the traditional security and privacy feedback loop with the user is severely hampered because voice is often the only input-output modality [15].

Independently approaching each of these deficiencies leads to incomplete protection against voice confusion attacks. First, while improving the speech interpretation process leads to fewer errors, there are fundamentally ambiguous situations that are impossible to resolve without any additional information (e.g., the difference between Fitbit and Phitbit). Second, Amazon's vetting of the skills is inadequate. While Amazon's guidelines for skill developers prohibits them from using phonetically-close skill invocation phrases, we demonstrate – similar to recent work [6, 12] – that attackers can violate these policies and get their skills published to the market [6, 12]. We are able to certify a "Phitbit" skill to squat on the Fitbit skill on the Alexa marketplace[1]. An attacker can also mirror the metadata of a legitimate skill, such as its invocation phrase, name, privacy policy URL and account linking URL implying that there is no distinguishing information between an attacker and legitimate skill. Voice assistant vendors could prevent skills with identical or phonetically-similar invocation phrases from being uploaded, but this limits flexibility and openness of the market and can lead to squatting behavior that is common on the web. For example, someone could register a skill for Fitbit and then ask the fitness tracking company to pay a large sum of money for that invocation phrase. There are also legitimate reasons for why skills have similar invocation phrases, such as multiple skills for the same task (Section 3.2). Finally, involving the user in the execution loop of each skill hinders usability — as discussed earlier, Amazon Alexa now provides auto-enabling of skills for enhanced usability.

We propose SkillFence, which adopts a systems-view of the problem and incorporates the user's preferences when speaking a voice command. Our key insight is that we can disambiguate skills with similar invocation phrases by analyzing the user's activity on *counterpart systems,* especially web and smartphone apps. Taking our example above, *Fitbit* has a website and a mobile app in addition to the Amazon Alexa skill. Most users will have already used these websites and smartphone apps before they turn to use skills. For some of them, the user has to sign up for the service through a website or smartphone app. Therefore, by noticing that a user has visited the Fitbit website consistently or has downloaded the corresponding Android app, we can select *Fitbit* (instead of *Phitbit*) as the correct skill when the user speaks the phrase "open Fitbit".

Achieving SkillFence's design requires overcoming several challenges: (1) Once we identify a user's counterpart activity, it must be *securely* matched to the corresponding Alexa skills. As we discussed, an attacker can mirror the metadata of legitimate skills. Thus, skills in the Alexa ecosystem do not have a notion of *secure identity* — a fundamental design principle in other computing systems (e.g., websites have TLS certificates and Android apps are signed). (2) Once we match a skill to its counterpart securely, we must ensure that only the matched skill gets invoked in response to an ambiguous phrase and despite the presence of phonetically-similar skills on the

[1]More discussion about the ethical issues around certifying this attack skill are in Section 3.2

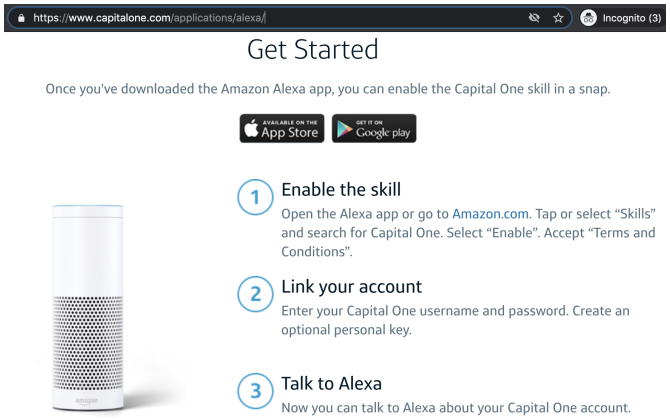


Fig. 1. Capital One skill information on its website. We extend the identity of the website to the Alexa skill if a link to it is found on a website we derive from the skill's metadata.

market. (3) We must achieve these two properties on the proprietary Alexa ecosystem over which we have no control so that we can protect users today.

SkillFence overcomes these challenges using a series of insights. First, we develop a secure identity for Alexa skills *without* modifying the proprietary platform. Our insight is that skill developers tend to link their Alexa skills directly from a website they own, much like they link their iOS and Android apps. For example, Capital One places a link to its Amazon skill on its website (Figure 1), like it does for its Android and iOS apps. Assuming that Capital One's website is not compromised, we obtain a secure link to the corresponding skill. Thus, we extend the user's trust in the website to the skill itself. Our system automatically discovers such links (Section 4.3.1).

The second challenge involves ensuring that securely matched skills execute in response to ambiguous commands. We discover that the Alexa backend provides two APIs — enable a skill and disable a skill — that can be used to control Alexa's invocation behavior. Specifically, we find that if a skill is enabled and its phonetic-neighbors are disabled, Alexa executes the enabled skill. We characterize and confirm this behavior using large-scale voice experiments (Section 5.3).

We implement SkillFence as a browser extension and a backend server. We evaluate it by designing and launching a data collection effort. Using the data of 116 users, our trace-based evaluation finds that SkillFence can select the correct skill (the one user intends to use) in 90.83% of the cases. We then evaluate SkillFence in the live Alexa environment by performing real time invocations for each skill used by the users. For those selected skills selected, SkillFence reduces the incorrect invocations from 17 to 0.

Finally, we provide an evidence-based set of design recommendations distilled from our experience in building and evaluating SkillFence. If implemented by stakeholders, they will secure the Alexa ecosystem and further improve the efficacy of SkillFence. For example, a skill vendor can place a link on its website that connects to the Amazon webpage listing of its skill, permitting SkillFence to automatically associate the identity of the website with the skill. We have started outreach efforts in explaining our recommendations to skill developers and are starting to see adoption — a skill developer has already updated their website to include a link to the Alexa skill, therefore helping SkillFence compute a secure identity for it (Section 6).

2 BACKGROUND AND RELATED WORK

2.1 Alexa Ecosystem

The Alexa ecosystem consists of a device placed in a user's home (e.g., smart speaker), a cloud platform implementing general voice assistant functionality, and an application endpoint implementing skill-specific functionality. The automatic speech recognizer (ASR) receives a command from the user and converts it to text. The natural language understanding (NLU) unit performs syntactic and semantic text analysis to determine the most appropriate skill matching the user's utterance. The skill developer hosts a cloud endpoint that implements the functionality API. For example, if the user utters *"Lyft, Get me a car"*, the ASR and NLU will eventually determine that Lyft can handle the command. Thus, the Voice Assistant cloud platform sends a message to the HTTP(s) API endpoint for Lyft. Skill developers provide the directory entry information, and they are in complete control of all aspects of that data. This includes the skill name, invocation phrase, suggested commands and URLs for privacy policies and account linking.

2.2 Skill Invocation

Amazon Alexa allows users to perform two skill operations - enabling and disabling. These operations can be executed via the skill's Amazon listing. Additionally, to promote usability Alexa automatically enables a skill when it is invoked by the user (e.g.,"Open Fitbit"). Once enabled, the skill can also be implicitly invoked (e.g., "Ask Fitbit how I slept last night"). Lastly, a currently disabled skill cannot be invoked until it is explicitly re-enabled.

2.3 Voice-based Confusion Attacks

Prior work has demonstrated the existence of frequently occurring and predictable errors in Amazon Alexa's speech recognition engine. These errors can be leveraged to develop malicious skills (with identical or similar invocation phrases) that can hijack the voice command meant for a legitimate skill. For example, the voice command saying "Alexa, open Fitbit," which is meant to invoke the *Fitbit* skill, but can trigger the malicious skill *Phitbit* after it has been published to the skill market. Upon activation, all voice interactions are handed over to the running skill which can perform a wide range of malicious activities like ask for private information or pretend to terminate and yet continue to operate by impersonating either other skills or the Alexa platform itself [28]. Kumar et al. introduced skill squatting attacks, where a malicious developer registers a phonetically similar sounding skill as the target [10]. Zhang N. et al. perform a similar analysis for Google Home, and introduce an additional attack where a fake skill masquerades as a true one [28]. Finally, Zhang Y. et al. introduce *lapsus* attacks that rely on common speech variation among humans [29]. For example, given a skill with invocation *"the true bank skill,"* a user might misspeak and instead ask for *"the truth bank skill."* An attacker can systematically discover common speech variations for a given phrase and then register skills. Broadly, all these attacks define a class of voice-based confusion attacks. The fundamental cause is the mismatch between a user's intention and the voice assistant's behavior.

2.4 Design Issues in Alexa Ecosystem

Lentzsch et al. recently performed a security analysis of the Alexa ecosystem that includes the vetting processes [13]. Like us, they find that attackers can circumvent the vetting and upload malicious skills to launch voice confusion attacks. They also observe that an attacker can mirror the metadata of legitimate skills, lending further confirmation that the ecosystem currently lacks any notion of secure identity. By contrast, our work contributes a method to provide a secure identity to skills through backlink search.

2.5 Existing Defenses

A direct way to combat voice-based confusion attacks is to prevent skills from being created with overlapping or identically-sounding invocation phrases. Kumar et al. suggest a phoneme-based analysis to detect similar-sounding skills [10]. Amazon Alexa includes developer guidelines that explicitly prohibit duplicate names [7]. However, these guidelines are not technically enforced. Recent changes to Alexa indicate that they now perform manual vetting, which is not scalable. Google Home's manual analysis detects when an invocation phrase is too close to an existing one. Although this seems like a reasonable approach, there are pitfalls. For instance, Google Home is prone to a phenomenon similar to domain squatting — a third-party developer scoops up invocation phrases that heavily overlap existing and popular services (e.g., Lyft, Papa John's), preventing the first parties from creating skills with those invocation phrases [9]. This also has the effect of third-parties benefiting from the copyright and reputation of the first-party invocation phrases on which they squat. In summary, although these techniques add defense-in-depth, they do not address the root cause of the problem.

Zhang N. et al. suggest a skill response checker that keeps track of the current skill's responses and computes a similarity score to other skills [28]. If two skills are found to be similar enough, an alarm is generated. Our work is orthogonal and represents a systems-oriented defense that securely predicts and enables skills using counterpart app information. We observe that a secure voice assistant platform benefits from both approaches.

Guo et al. built SkillExplorer, an NLP-based testing system that uncovers potentially policy-violating behavior [5]. By contrast, our work ensures that only the user-intended skills run in response to ambiguous utterances and does not pass judgment on whether the authorized skill violates policies.

2.6 Attacks on Speech Processing

A large body of recent work attacks voice assistants physically by injecting commands that exploit the ML models using adversarial examples [1, 2, 24] or the non-linear components of microphones [19, 26, 27] or the photoacoustic effect [21]. From SkillFence 's perspective, these are simply voice commands that come from a non-human source. Our work is independent of the origin of a command, and it is not intended to address physical attacks. Instead, our work focuses on disambiguating confusing voice commands by analyzing counterpart activity and then ensuring that the disambiguated command executes.

3 CHALLENGES IN PREVENTING VOICE CONFUSION ATTACKS

There are several fundamental design deficiencies that lead to voice-based confusion attacks. We identify them next and discuss how they guide the design of SkillFence.

3.1 Ambiguity in Speech Recognition

Natural language ambiguity is an unavoidable source of confusion for voice assistants. Homophones, homonyms, and various pronunciation intricacies have the potential to produce an error in the ASR component of the voice assistant [3, 5, 9, 11, 28–30]. For example, "Capital One" and "Capitol Won" sound identical, and in the absence of other information, it is not possible to identify the user's intended skill. Improvements in ASR accuracy will not solve this problem. Alexa could present a choice to users when confused; however, as discussed next, Alexa lacks the notion of a secure identity that prevents users from identifying which skill is legitimate, given a choice.

Objective 1: Seek a mechanism to resolve the ambiguity between phonetically similar skills that match a user's voice command.

3.2 Incompleteness of Metadata Vetting

Voice assistant ecosystems like Alexa vet the metadata of a skill before publication. This metadata includes the skill's name, account linking URL, privacy policy URL, developer URL, invocation phrase, and description.

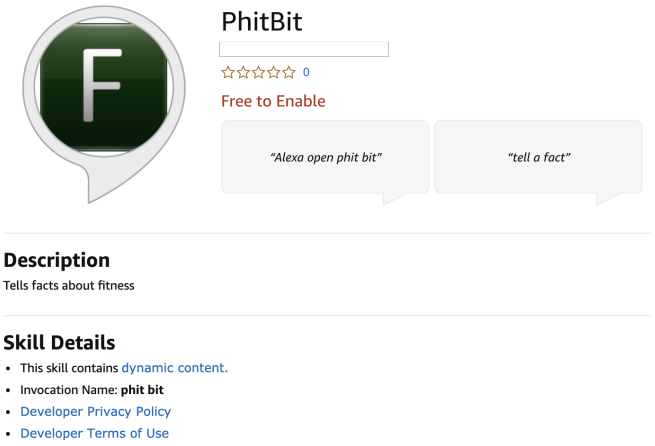


Fig. 2. Metadata attack on FitBit skill. Our attack skill targets the true FitBit skill including its account linking URL.

Recent work shows that an attacker can mirror the metadata of a legitimate skill and get it published to the marketplace [6, 12]. To an end-user, two different skills might appear identical to each other. As Alexa skills do not have a notion of secure identity, attackers can impersonate legitimate skills.

Implementing stricter vetting policies does not address this problem completely. There are valid reasons for skills to share metadata, such as the invocation phrases. For example, our market-scale analysis of Alexa skills finds that 7656 skills share their *name* with at least one other skill and 8978 skills share their *invocation phrase* with at least one other skill. If Amazon decides to prevent duplicate names, which of these skills to remove becomes unclear. Furthermore, requiring uniqueness in skill names can lead to a domain squatting effect — a third-party developer scoops up invocation phrases that heavily overlap with existing and popular services (e.g., Lyft), preventing the first parties from creating skills with those invocation phrases [9]. An alternative solution to this problem would involve prompting the users to decide which skills to invoke and then remembering their decisions. This approach requires user involvement (ideally through a screen), which goes against Amazon's rationale for automatically enabling skills.

3.2.1 Case Study. We developed the skill *PhitBit* with the invocation phrase `'phit bit'` that sounds similar to Fitbit, a popular fitness skill handling sensitive information. Fitbit, being an *account linking* skill, is required to provide an account linking URL in its metadata, which allows users to log-in and authorize it to access their Fitbit account. Figure 2 shows the Amazon Alexa skill page for PhitBit. When we submitted this skill for review, Amazon's vetting process prevented us from copying Fitbit's graphics, but it did *not* prevent us from using Fitbit's HTTPS API as the account linking URL nor Fitbit's privacy policy URL and developer URL. Our attack skill eventually passed the vetting process and was approved.[2] Multiple submissions were made to get the attack skill, *PhitBit*, approved. During one of the submissions, the vetting team informed us that the skill was not working because when they tried to invoke it, the command routed to the true FitBit skill instead of ours. On a subsequent submission, the voice command was routed to *PhitBit*. This incident shows that the vetting process itself is vulnerable to voice confusion. The certification of this skill violated multiple security testing policies described by

[2]Developers have two options while submitting a skill for review - (1) Certify and publish now, and (2) Certify now and publish later. We only submitted the custom skill for certification and never published the skill. Therefore, no users of Amazon Alexa interacted with the skill.

the Alexa skill review process. We disclosed this vulnerability to Amazon via the Amazon Vulnerability Research Program on HackerOne.
Objective 2: Establish the secure identity of a skill to prevent confusion with other skills.

3.3 Lack of Runtime Security Indicators

Once a malicious skill is invoked, it is easy for the attacker to deceive end-users — fake skill interactions mimic either the Alexa voice system or that of any other skill [9, 15, 17]. Most Alexa users are unable to differentiate between genuine and malicious skill interactions [15]. Unlike visual interactions, communicating security information over voice is not straightforward; it is hard to incorporate an HTTPS-like icon in a voice interaction. Once executed, a fake skill can ask the user for information that the user only shares with the target skill. For example, Find My Phone asks for a user's phone number, Transit Helper requests a home address, and Daily Cutiemails solicits a user's email address. This kind of attack can also damage the reputation of a legitimate skill, given that any poor service of the malicious skill will be blamed on the legitimate skill. Finally, the fake skill can lead to a phishing attack by delivering misleading information, such as a fake customer contact number, through the voice channel to the user. Thus, any defense strategy that involves the user interacting with a skill (after its execution) is insufficient.
Objective 3: Make the security decision (about the legitimacy of the skill) *before* a skill runs; only user-intended skills should execute.

3.4 Closed Skill Ecosystem

The code for an Alexa skill executes on the third-party developer's backend service that they fully control. Different from other ecosystems (e.g., Android), the skill code is not available to end-users or even Amazon. Thus, solutions based on OS changes to the Alexa device or code analysis techniques on the skill are not possible. Furthermore, the Alexa skill ecosystem is closed to incorporating built-in defense strategies from third parties, unlike the Android or web ecosystems.
Objective 4: Develop the defense strategy to be practical for deployment without modifications to Alexa.

4 SKILLFENCE DESIGN

In the following, we describe the design of SkillFence, starting with the system and threat model, followed by an operational overview. We then elaborate on the individual components of SkillFence.

4.1 System and Threat Model

We assume a malicious skill developer that launches voice-based confusion attacks [10, 28, 29]. These attacks depend on the user's inability to identify the precise skill they are interacting with and on errors in the speech processing components. The attacker's goal is to get their malicious skill executed so that they can use it to conduct phishing or execute other restricted actions [10, 28, 29]. As the current Voice Assistant model does not properly vet skill metadata (Section 3.2), an adversary can upload malicious skills with arbitrary metadata, including metadata that is directly copied from legitimate skills. We assume that the end user's devices are trustworthy: the voice assistant itself, smartphones, and web browsers. We also assume that the websites of the legitimate skill developers are not compromised. For example, the Fitbit or the NBC website is not compromised and is trustworthy.

4.2 Operational Overview

Skill invocations are not equally ambiguous. For example - the "Fitbit" skill is much more likely to be *confused* with the "Phitbit" skill rather than the "Lyft" skill. For skills that are phonetically ambiguous, SkillFence uses

information from the user's history to nudge the Voice Assistant towards executing the skill that matches the user's preferences — thereby resolving the confusion. There are four questions associated with SkillFence's operation: (1) *how to identify skills that are phonetically similar?* (2) *how to identify the user's preferences?* (3) *how to match the user's preferences to a skill?* and (4) *how to nudge the Voice Assistant to execute the matched skill?* In the following, we discuss – at a high level – how SkillFence's design answers these questions while achieving the four objectives from the prior section.

4.2.1 Phonetic Similarity. We model the phonetic similarity relationship of skills as a phonetic graph. In this graph, skills are the vertices, and the edges represent skills that are phonetically similar, i.e., potential targets for voice confusion. SkillFence uses this graph to reduce the potential of voice confusion attacks.

4.2.2 Collecting the User's Preferences. An end-user installs the browser extension that we have built, and it periodically processes counterpart information — browser history and Android apps (that the extension collects by navigating to the play.google.com website). This counterpart information represents the user's preferences (**objective 1**). For example, assume the user has purchased a Fitbit device. It is likely that they will have accessed Fitbit websites or installed the corresponding Android app. Later on, they might start using the Fitbit Alexa skill. The SkillFence extension processes the prior URL visits or Android app installation information and encodes this information in the user's phonetic graph to resolve the confusion at run-time. Thus, when the user speaks the command "talk to Fitbit," SkillFence will know they mean the fitness tracking skill and not some other neighboring skill in the phonetic graph. We note that browser history and android app collection occur in a privacy-respecting way. This data is never transmitted outside the extension; rather, matching browser history to skills occurs locally. Furthermore, we implement existing techniques to ensure that our system is not tricked by web history poisoning (see Section 5.5.1 for more details).

4.2.3 Securely Mapping Preferences to Skills. Matching the counterpart activity to the Alexa skill requires a secure skill identity (**objective 2**). However, as we have discussed, the Alexa ecosystem does not have such an identifier. A contribution of our work is a practical way to achieve secure identity by only requiring small changes on the skill developer's websites. For a given skill, the SkillFence backend extracts all URLs from metadata (e.g., account linking URL, privacy policy URL, developer URL). It then searches through these websites for a link *back* to the Amazon Alexa marketplace listing for the skill. If such a backlink exists, then we associate the identity of that website (i.e., HTTPS certificate) with the skill. In our running example, SkillFence maps the user's visits to fitbit.com to the correct Fitbit skill.

4.2.4 Practically Ensuring that Only Matched Skills are Executed. After SkillFence has identified and securely matched the user's preferences to a skill, it must ensure that Voice Assistant executes the matched skills, despite confusing voice commands and the presence of possible attacker skills (**objective 3**). In our example, SkillFence must ensure that the Fitbit skill executes instead of the Phitbit skill. Furthermore, SkillFence must achieve this task on the existing Alexa system without modifications (**objective 4**). Our work overcomes this challenge by discovering and characterizing a feature in Alexa skill execution: given two identical skills, if one of them is in the *enabled* state while the other is in the *disabled* state, Alexa always executes the enabled one. SkillFence leverages this behavior to explicitly enable the matched skills and disable their neighboring skills – nudging Alexa towards executing the matched skill. Back to our example, SkillFence enables the correct Fitbit skill and disables the Phitbit skill (as well as the other neighboring skills). When the user speaks the command "open Fitbit," Alexa will not be confused with the Phitbit skill and will execute the correct Fitbit skill. Section 4.3 elaborates more on this aspect, which we empirically characterize in Section 5 using large-scale voice experiments.

4.2.5 Putting Everything Together. SkillFence consists of a user-facing extension and a backend component as shown in Fig. 3 The backend component of SkillFence periodically scrapes the Alexa skill metadata. It performs

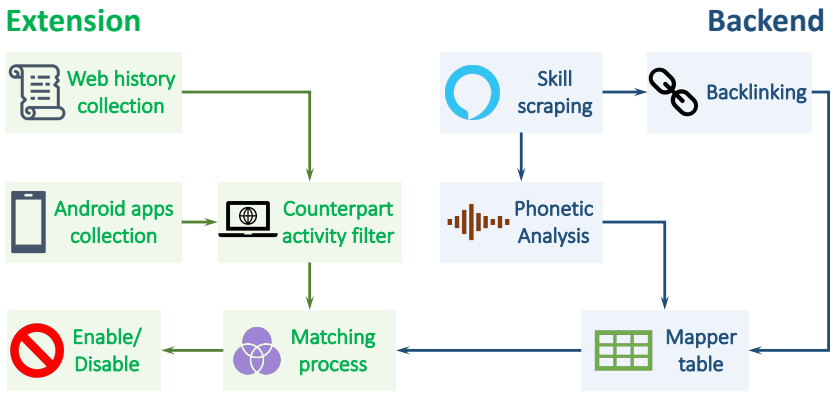


Fig. 3. End-to-end system overview diagram.

backlink analysis to establish a secure skill identity. It also models the phonetic similarity among skills as a phonetic graph. SkillFence's extension periodically collects the user's preferences from counterpart activity and securely maps them to a set of skills. It then uses the phonetic similarity graph to enable/disable the appropriate skills to reduce voice confusion attacks. In particular, SkillFence's extension enables skills that match the user's preferences from counterpart activity and disables their neighbors. The design of SkillFence achieves **objective 4** as it requires no change to either the skills or the Alexa ecosystem; it leverages existing tools to meet the first three objectives. In the next two sections, we discuss each of the above steps in detail, outlining the underlying algorithms and data structures.

4.3 Backend Component

The backend component of SkillFence periodically analyzes the skills in the Alexa marketplace. It performs two operations: (1) secure identity generation using backlink analysis and (2) phonetic graph construction for modelling skill phonetic similarity (which the extension eventually uses to enable/disable skills). The backend distills the results of these two operations into a *mapper table*.

4.3.1 Secure Identity Generation. The key insight for generating secure identity is to leverage trust in the existing PKI and web ecosystem. SkillFence searches through the domains in a skill's metadata for a URL linking back into the Amazon marketplace listing for that skill. Assuming that the malicious-skill attacker in our threat model has not also compromised the legitimate skill's website, the presence of the *backlink* indicates that indeed, the developer of the website and of the skill are the same entity, and thus, the identity of the skill is the identity of the website (i.e., the TLS certificate). Figure 4 illustrates the high-level idea, for the NBC skill as an example. SkillFence backend will extract all metadata URLs and search for a URL to the NBC Alexa skill page. Once found, it will associate NBC's website identity with the skill in the mapper table data structure.

Concretely, the backend extracts URLs from each skill's metadata, including the privacy policy, the developer website, and the account linking URLs. Then, it extracts the root domain of each URL to search for webpages that link to the skill. In particular, SkillFence issues Google search queries of the type - site: <domain> "alexa skill." Then, it uses Selenium to crawl the webpages for the top 100 search results, while searching for the skill's Amazon URL. When it finds such a link, SkillFence establishes a mapping between the public key certificate (if available) of the domain and the skill. SkillFence also uses link backtracking services [4] to find whether any of the websites from the skill's metadata link to the skill's Amazon page.

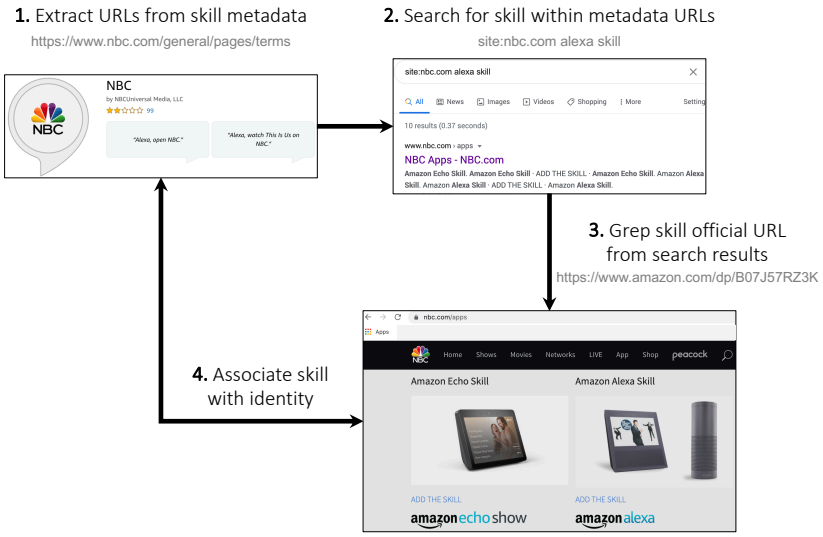


Fig. 4. Generating secure skill identity: SkillFence searches for backlinks to the same skill's Amazon listing within domains that are extracted from the skill's metadata. If there is a link from a domain back to the skill, then the skill-domain pair is added to the mapper table.

This method presents exhibits two advantages. First, because it is automated, it scales over a large number of skills in marketplace. Second, it allows skill developers control over improving security for their skills rather than relying on Amazon's vetting process. Compared to alternatives, such as updating the voice assistant platform to support cryptographic signatures, this design requires a simple change on skill developer websites; they need to only include a backlink to the Amazon marketplace listing of the skill. This also serves to improve the credibility of the skill as Android users too are recommended to rely on official websites of popular services like Facebook, WhatsApp, Truecaller to find trusted sources for links to the apps [18]. We have begun explaining the benefits of this approach to skill developers and convinced one developer to update their website with a link to their skill (*OurGroceries*[3]), demonstrating the value of our approach. An entity like Amazon can easily require skill developers to implement this change.

4.3.2 Phonetic Graph. SkillFence uses the phonetic graph to identify potential targets of voice confusion. To construct the graph, we first model the phonetic similarity between skills using a distance metric. This metric is used to derive a weighted graph where the vertices are skills and edges are weighted by the phonetic distance. SkillFence uses this graph to generate a list of phonetic neighbors for each skill, which are added to the mapper table.

Phonetic Distance. We define a distance metric that quantifies the confusion between two skills based on the phonetic representations of their invocation phrases. This metric ensures that different phoneme transformations have different costs. To achieve this property, we compute a weighted Levenshtein distance between the phonetic representations where different phoneme edit operations have different costs. We compute the weighted cost matrix by applying the Needleman-Wunsch Algorithm on the CMU dictionary, which has around 9181 pairs of

[3]Skill - https://www.amazon.com/HeadCode-OurGroceries/dp/B01D4F1J0M
Website - https://www.ourgroceries.com/user-guide#installing

alternate pronunciations (also used in prior work [28]). For example, the substitution cost for replacing phoneme α with phoneme β:

$$WC(\alpha, \beta) = 1 - \frac{SF(\alpha, \beta) + SF(\alpha, \beta)}{F(\alpha) + F(\beta)},$$

where $F(\alpha)$ is the frequency of occurrence of phoneme α and $SF(\alpha, \beta)$ is frequency of occurrence where phoneme α has been substituted by phoneme β in all alternate pronunciation pairs in the corpus. This cost function ensures that phoneme substitutions that provide valid alternate pronunciations of English words are given a smaller weight. We also normalize this distance by the length of the invocation phrases.

Constructing the Phonetic Graph. We utilize the phonetic distance to compute the weights of the edges in the phonetic graph. We then prune the graph by dropping all the edges with distance greater than a threshold; the tuning procedure of the threshold is discussed in Section 5. The edges in the resulting phonetic graph represent instances that are prone to confusion attacks. The skills with no neighbors are less prone to speech interpretation errors. As an illustration, Figure 5 shows the graph for six skills, represented by their invocation phrases. For a phonetic distance threshold of 200, the graph has only four edges (represented as solid lines). In this example, the nodes with thick borders represent the skills ("nicole facts", "Fitbit") matched using counterpart activity collection. The extension component enables the matched skills (colored in green) and disables their neighbors (colored in red). As we explain later, this process reduces the number of incorrect invocations due to voice confusion (e.g., between “Fitbit” and “Phitbit”). Skills not connected to an enabled skill need not be disabled. We evaluate the effectiveness of this approach in Section 5.3.

Trade-off. SkillFence aims to disable all skills connected to an enabled skill in the pruned phonetic graph. A smaller distance threshold would lead to fewer edges, resulting in an efficient disabling process. However, it increases the chances of incorrect invocation between unconnected skills. A sufficiently large threshold can reduce confusion, at the cost of having to disable a larger number of skills and thus, a less efficient disabling procedure. It also can increase the potential of disabling skills that the user might later need. We empirically tune the threshold to obtain a reasonable trade-off between potential for invocation confusion and efficiency of disabling at runtime (Section 5.3.4).

4.3.3 Mapper Table. Each row of the mapper table contains an Amazon marketplace skill URL, the domain and its certificate that forms the secure identity, and a list of neighbor skills (derived from the phonetic graph). The extension uses this information at runtime to index and enable/disable skills. The backend process continually updates the mapper table and the extension periodically downloads it (approx. 5 MB for the initial download).

4.4 Extension Component

The browser extension of SkillFence is its user-facing component. It interacts periodically with the backend component to fetch the up-to-date mapper table. Also, it *locally* extracts the user’s activity with counterpart apps to enable the skills that match the user’s activity. Finally, it uses the mapper table to identify and disable the skills within a phonetic distance from the enabled skills.

4.4.1 Counterpart Activity Collection. SkillFence runs as a Chrome-based browser extension. The extension periodically (every 5 minutes) retrieves user’s installed app data and browser history from their Google account. It scrapes the user’s “My Android apps” page on play.google.com/apps to obtain the list of installed Android apps. For browsing history, it uses the *chrome.history* API to query the browser records. The extension only searches for URLs whose root domains are part of the Mapper Table. There are three considerations associated with this process: execution overhead, user privacy, and the potential for adaptive attacks.

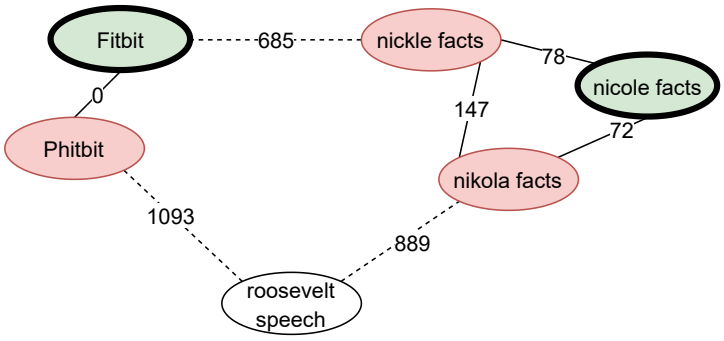


Fig. 5. Phonetic graph of a set of skills with phonetic distance between their invocation phrases as weights. The skills ("nicole facts", "Fitbit") are mapped using counterpart activity. Note: the dashed lines are dropped edges. Green/Red nodes represent skills that are enabled/disabled by SkillFence respectively.

First, the extension periodically scrapes the installed apps to identify the newly installed apps. Further, it collects the history using browser APIs, which do not incur any noticeable overhead (<5 sec). Second, this process takes place *locally*, and the collected information never leaves the user's browser. Third, an attacker might exploit this process to poison the user's browsing history. Without special attention to this case, SkillFence could potentially use visits to attacker-controlled websites to enable attacker-controlled skills.

SkillFence counters the web poisoning attacks through a web filtering heuristic. It only considers websites with the following properties: (1) the website is visited across multiple valid sessions, (2) each valid session consists of visits to at least three different pages within the website's domain, (3) the considered visits are not directly preceded by visits to skill pages (these criterion are based on past work)[14, 22, 31]. The first two criteria focus on websites that the users are likely to visit without attacker influence through phishing or spamming. The third criterion prevents SkillFence from recording visits to a website that are followed by visits to the skill pages.

Trade-off. The selectivity of the web history filter can be adjusted to address the trade-off between usability and security. A more selective filter makes it harder for the attacker to get a malicious skill enabled, but it would also lead to the collection process missing out on genuine websites that the user actually visited. Whereas, a filter that considers most of the visits is likely to have a large coverage but would be easier to evade.

4.4.2 Matching Process. After URL filtering, SkillFence matches the user's visited websites or installed apps to Alexa skills. SkillFence obtains the list of installed apps and the user's browser history. It then extracts the certificate from each visited website and installed app. Using the pre-fetched mapper table and these certificates, it matches the visited websites and installed apps to their corresponding skills. Recall that the mapper table stores associated certificates for each skill. This process leaks no information about the user's history of activity with counterpart apps. The matching is performed locally with the entries of the mapper table.

4.4.3 Enable/Disable Module. The final step is to enable the matched skills and disable their phonetic neighbors. Amazon's skill listing interface allows a user to perform two basic operations: *enable* and *disable* skills. We empirically observe (Section 5) that enabling a skill and disabling others with similar invocation phrases ensures that Alexa invokes the enabled skill and thus, prevents confusion attacks.

SkillFence leverages this observation to enable all skills that were matched using the mapper table and user's history. Next, it disables the phonetic neighbors of the skills enabled in the previous step. The extension performs these operations on behalf of the user. Since the disabled skills cannot be invoked without explicitly enabling them first, this also acts a warning mechanism for incorrect invocation.

4.4.4 Online Operation. As discussed earlier, SkillFence is a Chrome browser extension. To run it, the user needs to: (1) install the extension on their Chrome browser; (2) allow the extension to access their web history and/or installed apps; and (3) allow the extension to enable/disable skills. If the user does not provide the required permissions, then SkillFence will not be operational.

At installation time and each hour thereafter, SkillFence communicates with its backend server to fetch the latest mapper table. During its initialization on a new device, SkillFence fetches the the complete mapper-table of size 5MB. Subsequent transmissions include delta updates (2KB per skill). Assuming updates to around 26 skills everyday [25], this results in a total network overhead of 50KB/day. SkillFence retrieves the counterpart data and uses the latest mapper table to identify the matched skills. Then, SkillFence applies the Enable/Disable module for the matched skills.

SkillFence does not maintain per-device or per-account history. Given counterpart data, it enables/disables skills based on their current state in the user's Alexa account and does not interfere with previously-enabled skills. The counterpart data can come from different sources, such as web history, app installations on the same or different devices. Thus, the user can install SkillFence on multiple devices, and even configure different google accounts (for counterpart activity) against a single Amazon Alexa account. Despite this diversity of counterpart data sources, SkillFence operates with a single Alexa account at any given moment, and thus, it has an accumulative effect on that Alexa account even if the user has multiple computers (e.g., desktops, laptops, smartphones).

Finally, SkillFence might not pre-enable some legitimate skills that the user wants to be executed. This could happen because there is not enough counterpart activity or the skill does not have a secure backlink. In such a case, SkillFence does not affect the user experience, it defaults to the state of the world today: Alexa will pick a skill according to its matching algorithm. As such, SkillFence does not deteriorate the security of Alexa users, but might harm their user experience when it disables skills they might need in the future. We discuss these trade-offs in the evaluation.

5 EVALUATION

We perform an end-to-end evaluation of SkillFence and it's core components. We summarize our evaluation findings, then discuss the experimental setup and finally provide experimental details. We also discuss how SkillFence's design protects against an adaptive attacker with knowledge of its inner workings.

5.1 Evaluation Overview

Our evaluation answers the following questions:

Q1. ***What are the characteristics of Alexa Enable/Disable API in controlling which skills get executed?*** We perform a large scale measurement to test SkillFence's Enable/Disable module against phonetically similar skills available in the Amazon Marketplace. These skills include ones that share invocation phrases or have similar sounding invocations. We establish that when a skill is enabled and its phonetic-neighbors are disabled, Alexa always executes the enabled skill.

Q2. ***How effective is SkillFence in preventing voice confusion attacks? How does SkillFence balance the trade-off between security, usability, and performance?*** We perform a trace-based evaluation with real user data from our data collection effort. For a distance threshold of 400, SkillFence is able to perform at a False Rejection Rate of 9.17%, a False Acceptance Rate of 19.83% and an Equal Error Rate of 16.5%. Our evaluation also shows that the initial setup time depends on the phonetic distance threshold. The optimal threshold between usability and security corresponds to one hour of initial setup time. This initial time is a side-effect of implementing SkillFence with a closed system; a better API design from the manufacturer can significantly cut this time down. Subsequent SkillFence

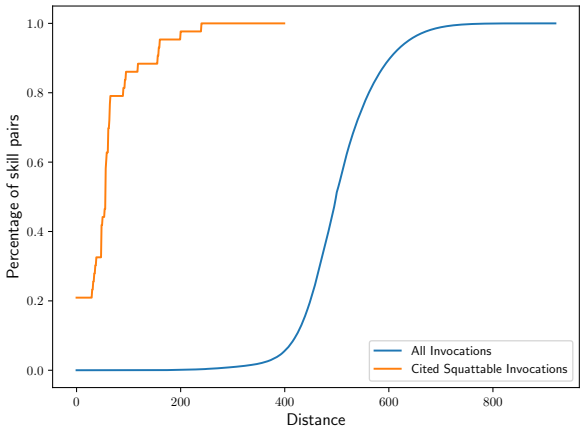


Fig. 6. CDF of phonetic distance between invocations of all possible Skill pairs. More than 95% skill pairs have phonetic distance larger than 400. All skill squatting examples cited in previous work lie within a phonetic distance of 250.

operations encounter fewer enabled skills at a time, leading to a much quicker response time of less than a minute.

Q3. ***How does SkillFence protect against an adaptive adversary? What are the privacy implications of using SkillFence?***
The different components of SkillFence are designed to address an adaptive attacker and make it more challenging to launch a successful skill-squatting attack. SkillFence does not maintain any state or store any user information, even locally, thus minimizing any privacy concerns.

5.2 Experimental Setup

First, we perform a large-scale measurement to characterize the phonetic similarity thresholds of Alexa and confirm its enable/disable behavior. Second, we perform trace-based evaluation to study the user experience of a SkillFence user. We collect the counterpart history and the enabled skills from a set of Alexa users. We use the collected data to mimic the user experience with SkillFence. First, we initialize the backend of SkillFence by populating the Mapper Table. Then, we use the counterpart history data of each user to enable/disable skills on their phonetic graph. We create a test Amazon account for each user and populate it with their phonetic graph; we then invoke skills from the user's reported history as well as their phonetic neighbors. Effectively, we characterize the security gains as well as the usability and performance overhead for each user.

We focus our analysis on a subset of the Amazon Alexa skill Marketplace: the 3659 skills with account linking, which represent sensitive skills on the Alexa market according to a recent analysis by Shezan et al. [20]. Account linking skills are likely to be the targets of attack because of their access to the user's account data. We designate skills that have account linking as critical. Note that SkillFence's design, implementation, and evaluation can be readily applied for the full set of skills without changes to the Alexa Marketplace or how skills operate on the Alexa platform.

5.2.1 SkillFence Backend. We generate the Mapper Table for the 3659 Amazon Alexa skills with account linking. This process includes running the backlinking analysis to extract the secure links of the skills and constructing the phonetic graph.

Backlinking. The majority (2496 out of 3659) of skills contain at least one valid unique domain in their provided metadata. Skills that share domains generally belong to the same developer (we do not consider domains of popular cloud hosting services such as Amazon and Google, for backlinking). A skill-domain pair is part of the Mapper Table only if a website associated with the domain lists a link to the skill's Amazon homepage. To search for backlinks, we follow two approaches – Google Crawling and Link Backtracking Services. For Google Crawling, we constructed queries like "site:<domain name> Amazon Alexa Skill" and searched through the top 100 results. We also used additional text phrases such as 'Alexa Skill,' and 'Amazon Skill,' to exhaustively search for all Alexa skill references. These queries provided at least one search result for 2023 skills. Among these, we found backlinks for 404 skills. For building a more exhaustive backlink profile, we utilize a Search Engine Optimization (SEO) service, *Ahref*[4], which provides a tool for retrieving all backlink references for a given URL. Overall, employing both approaches, we were able to find the backlinks for 474 skills. This number is likely to increase as websites start adding references to their skill's homepage like they currently do for their Android and iOS apps. Android users are recommended to rely on official websites of popular services like Facebook, WhatsApp, Truecaller to find trusted sources for links to the apps [18]. The secure identity generation process automates this process on the user's behalf. Furthermore, as we discuss later, we have begun outreach efforts to encourage skill developers to include a link to their Alexa skills on their webpage, and we are beginning to see adoption.

Phonetic Graph. We constructed the skill phonetic graph consisting of 51964 skills. Figure 6 shows a CDF of weight values of edges in the Phonetic Graph, i.e., the phonetic distance between all pairs among 51964 skills. The mean distance is around 500 and most of the distance values are less than 800. We also observe that only 5% of the weights lie below the value of 400. This suggests that pruning all edges with weights more than 400 would lead to a sparse phonetic graph and considerably reduces the number of skills to be disabled. This is important because it improves SkillFence's efficiency. Figure 6 also shows that all skill squatting invocation pairs introduced in prior work have phonetic distances less than 400 [10, 28].

5.2.2 Data Collection. We performed a data collection campaign to collect installed mobile apps, skill-relevant browsing history, and preferred Alexa skills from a set of individuals through Amazon Mechanical Turk (MTurk). This data allows us to perform trace-based evaluation of SkillFence's extension component. We focused the data collection on individuals who own Alexa devices and Android smartphones. We obtained IRB approval for the data collection procedure from our institution.

Data Collection Design. We designed a Qualtrics survey hosted on MTurk to guide individuals through the data collection procedure. The survey consists of five sections: eligibility, Chrome extension installation, data collection, data validation, and clean-up. After completing a consent form that verifies age, Alexa ownership, and records the MTurk ID, the participants install a Chrome extension and log into their Amazon and Google accounts. The users enter their MTurk IDs again in the extension as a verification check.

The data collection procedure involved two components: (1) extracting user data consisting of enabled skills, browsing history, and installed Android applications; and (2) validating this information through a survey form. We created a Chrome extension to automate data extraction. This extension scrapes the enabled Alexa Skills from a user's "Your Skills" page on *alexa.amazon.com*. It also scrapes the user's "My Android apps" page on *play.google.com/apps* to obtain the list of installed Android apps. In the browsing history, the extension only searches for URLs whose root domains are part of the Mapper Table — other URL data is not transmitted to our servers.

There are two potential problems with the extracted data. First, the user could have cleared their browsing history. Second, a user might have enabled a skill with which they do not interact. Note that the first issue applies

[4] https://ahrefs.com/backlink-checker

only for the data collection and does not affect SkillFence as it runs continuously at the user's side. To address these problems, the survey populates a list of domains (domains that are mapped with any of the user's enabled skills) and asks the user to select all domains that they would have visited.

For the second issue, we define *used skills* as the enabled skills the user frequently interacts with. To address potential over-represented skills in the data collected (those enabled but not used), we ask the user to validate the extracted data. The survey also populates all the extracted enabled skills, along with each skill's title and a screenshot of the skill's Amazon homepage, and asks the user to select all skills that they interact with. For both the questions, additional random records (20% of the total records) are added to the populated list. This acts as an attention check for the user. Any user that selects the additional records is removed from the analysis. When the survey detects that it has received the list of apps and skills using the participant's MTurk ID, it guides them to uninstall the extension.

Data Collection Procedure. We recruited 116 individuals through Amazon's MTurk who satisfy the eligibility criteria of age and device ownership. We did not enforce any additional qualifications because the survey has built-in safeguards that ensure the participants correctly installed and executed the extension. We paid each participant $3 for completing the survey; 95% of the respondents finished the survey within 9.7 minutes. The rest of the participants faced some technical issues while installing the extension, and we helped them through the process using email. This communication did not affect the anonymity of the data. Finally, we collected the list of installed Android apps, visited web domains, and preferred skills from these 116 individuals.

We conduct a preliminary check to test whether all the skills from the data collection were included in the Mapper Table. We found that 72 skills out of 162 skills were not present as their backlinks were not listed on the websites. We manually verified these skills and added entries to the mapping table.

5.2.3 Runtime Skill Invocation. We perform a runtime evaluation by invoking each skill on a test Amazon Alexa account. This evaluation serves three purposes: (1) perform a large-scale measurement to characterize the phonetic similarity between the skills, (2) confirm Alexa's enable/disable behavior, and (2) assess the security gains for each user in the collected data. We designed an automated and systematic setup to invoke each skill of interest using organic and synthesized speech. We feed an invocation audio command to the Alexa Voice Service to directly interact with the Alexa backend. This invocation command is of the form: "Alexa, open <skill invocation phrase>." We terminate the interaction with the Alexa backend once the skill invocation occurs or if Alexa cannot process the request. We classify the result of each invocation as: "correct," "incorrect," or "no invocation." The last category occurs when Alexa fails to run any skill.

We create invocation audio using human speech reconstructed from the LibriSpeech dataset [16] and Google TTS (Text-to-Speech service[5]). LibriSpeech consists of transcripts for around 1000 hours of spoken English from 2484 speakers. To construct an invocation audio from this corpus, we first search for any continuous utterance of the invocation phrase. If nothing is found, we break the invocation phrase into smaller components and stitch together the respective audio clips to create a full phrase. We ensure that the components were derived from the same speaker. Since some skill invocation phrases contain non-English words (e.g., Amex), we were not able to generate organic invocation audio for all skills in the dataset. To avoid any speaker-specific errors, we perform each trial with audio samples collected from three different speakers. Then, we take the majority result from the three outcomes.

5.3 Q1. Enable/Disable Module Evaluation

SkillFence relies on the Alexa enable/disable API to ensure that the matched skills execute despite the presence of phonetically-close skills. We perform large-scale voice experiments to establish that, when enabling a matched

[5] cloud.google.com/text-to-speech

Table 1. The invocation outcomes for skills with *identical* invocation phrases. For the default state, Alexa incorrectly invokes skills. When enabling the target skill and disabling its phonetic neighbors, Alexa has no incorrect invocations.

	Text-to-Speech Experiment	
Metric	Default State	Target Enabled, Others Disabled
Correct Invocation	2	23
Incorrect Invocation	5	0
No Invocation	1	5
	Audio Recording Experiment	
Correct Invocation	2	21
Incorrect Invocation	2	0
No Invocation	4	7

skill and disabling its phonetic neighbors, Alexa always executes the enabled skill. These experiments also inform the distance threshold needed to identify phonetic neighbors.

5.3.1 Skill Datasets. The high-level approach is to repeatedly invoke sets of phonetically-close skills and observe the results. We created the following two datasets:

Dataset of skills with identical invocation phrases. We create 8 disjoint sets of skills where each set contains members with identical invocation phrases. The complete list of skills is in Appendix A. Note that each skill is randomly taken from the Alexa marketplace (any repetitions in skill names and invocation phrases occur naturally). We have a total of 28 unique skills. We constructed these sets to test SkillFence's Enable/Disable module against skills that share invocation phrases with different number of other skills.

Dataset of skills with similar sounding invocations. We select the largest set of skills with unique invocations which satisfies the property — for each skill, the set also contains at-least two other skills whose invocation phrases are at a phonetic distance of less than 600. This set of 53 skills represents a phonetically dense cluster of skills. We select these skills from the whole Alexa market rather than just the set of critical (i.e., account linking) skills to find the ones that are most vulnerable to voice confusion attacks.

5.3.2 Results for Identical Invocation Phrase Dataset. We use the Google TTS API and reconstructed human audio to generate audio for invocation phrases, and run these experiments.

(1) Baseline: With all skills in default state, play audio for invocation phrase. Our intended target is the most popular skill (with most reviews) in the set.
(2) SkillFence: For each member of the set, enable that member, disable the rest of the set's members, and play audio. The intended target is the enabled skill.

For the SkillFence scenario, we report outcomes for 28 trials while enabling each skill and disabling the rest. However, for the Baseline case, as all the skills are in default state and all skills in one set have identical invocation phrases, we only perform 8 trials, one for each set. Table 1 contains the results. We assign the the most popular skill (with most reviews) in the set as the intended target because popularity is a factor in deciding skill invocation [17].

Baseline: For TTS audio, we observe that in 2 cases, the user's intended skill is executed ('correct invocation'), in 5 cases, the unintended skill is invoked and in 1 case, invocation fails. Similarly, for LibriSpeech audio, there

Table 2. The invocation outcomes for skills with *similar sounding* invocation phrases. For the default state, Alexa incorrectly invokes skills. When enabling the target skill and disabling its phonetic neighbors, Alexa has no incorrect invocations. Note that the number of trials varies between TTS and Audio Recording as not all of the invocation phrases are in the LibriSpeech dataset.

	Text-to-Speech Experiment	
Metric	Default State	Target Enabled, Adversary Disabled
Correct Invocation	62	96
Incorrect Invocation	30	0
No Invocation	14	10
	Audio Recording Experiment	
Correct Invocation	25	46
Incorrect Invocation	13	0
No Invocation	10	2

are 2 cases of 'correct invocation', 2 cases of 'incorrect invocation' and 4 cases where invocation fails. Both the experiments conclude that in the presence of identically named skills, confusion does occur.

SkillFence: For TTS audio, correct invocation of the user's intended skill occurs 23 times when the intended skill is enabled and unintended ones are disabled (5 result in invocation failure). For LibriSpeech audio, there are 22 correct invocations and in 7 cases invocation fails. There are no incorrect invocations when the intended skill is enabled and others are disabled. This trial confirms our observed behavior of the enable/disable API — when the intended skill is enabled and phonetically-close skills are disabled, only the intended skill executes in response to an ambiguous invocation phrase.

5.3.3 Results for Similar Invocation Phrase Dataset. In this experiment, we conducted the invocation trials by selecting pairs of skills (A, B), such that skill B is the phonetically closest one to skill A. By design, there are 106 such pairs when we use Google TTS and 48 such pairs when we use reconstructed human audio. For each pair, we test the invocation of skill A in two configurations:

(1) Baseline: Default State (skill A disabled, skill B disabled)
(2) SkillFence: Target Enabled, Potential Adversary Disabled (skill A enabled, skill B disabled)

Configuration (1) is the control condition that tests how Alexa responds to confusing voice commands currently. Configuration (2) simulates the case where SkillFence predicts that skill A is the user's intention.

Table 2 shows the skill invocation results for both GTTS and organic audio. Configuration (1) incurs 28% and 27% incorrect invocations respectively for the two modes of audio. For configuration (2), the number of incorrect invocations is zero. Thus, we conclude that the behavior of Alexa supports our hypothesis: when a skill is enabled and its phonetic neighbors are disabled, the voice assistant will automatically execute the enabled one.

5.3.4 Enable/Disable Threshold. For the baseline configuration of the similar invocation phrase experiment, there were 30 and 13 incorrect invocations for the TTS and human speech trials respectively. Figure 7 shows the phonetic distance between the invocation of the desired skill and the incorrect skill that was actually invoked. Incorrect invocation occurrences were more frequent between skills whose phonetic distance was smaller. Also, no incorrect invocation occurred between skills with phonetic distance more than 400. Therefore, for the current set of registered skills on Alexa, SkillFence, when configured with a threshold greater than 400 will be able to

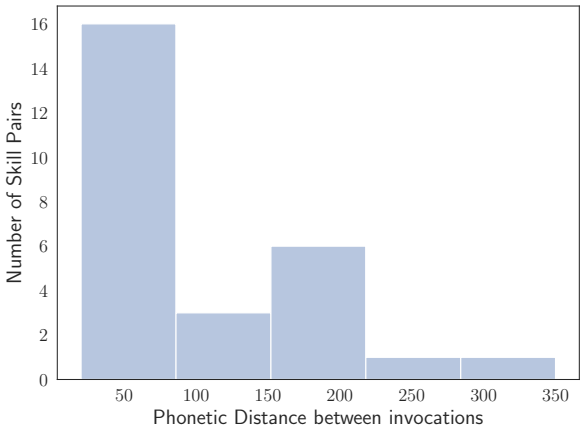


Fig. 7. The distribution of phonetic distance between all skill pairs that resulted in incorrect invocation. There are no incorrect pairs beyond a distance of 400.

prevent voice-confusion attacks between any skill pair (assuming the target skill has been mapped). Hence, we fix the phonetic distance threshold for SkillFence to be 400 for subsequent experiments.

5.4 Q2. End-to-End SkillFence Performance

We perform a trace-based evaluation of SkillFence to study its end-to-end performance in terms of the effectiveness in preventing voice confusion attacks, impact on usability, and performance overhead. For each user from our data collection campaign, we generate a list of critical matched skills by running the extension module on the collected counterpart data. Recall that the set of critical skills are ones that are likely to be squatted upon [20]. This process involves extracting certificates from the visited websites and installed Android apps and then matching the skills from the Mapper table. We create a test Amazon account for each user, enable the matched skills, and disable their phonetic neighbors. We compare the enabled and disabled skills (i.e, those matched from counterpart data and their neighbors) to the skills actually used by the users (reported in the data collection).

5.4.1 Usability vs Security. The phonetic distance threshold controls the trade-off between usability and security in SkillFence. A lower threshold corresponds to a sparser phonetic graph, implying less skills to disable for each matched skill. This setting favors usability as the user can set up SkillFence faster and can interact with a larger number of skills at the cost of potential incorrect invocations. Whereas, a larger threshold results in a more connected graph with a larger number of skills to be disabled for each matched skill, reducing the invocation confusion and improving security.

We quantify the impact of the phonetic distance threshold through two types of errors: False Rejection Rate (FRR) and False Acceptance Rate (FAR). FRR measures the probability that SkillFence disables a skill that the user actually needs — representing a proxy for usability. FAR measures the probability of SkillFence missing a squatting skill and not disabling it — representing a proxy for security. Recall that in the previous section we show that a pair of skills with a phonetic distance of less than 400 can lead to an ambiguous invocation. Using this empirical measure, we consider unused skills, with a distance of less than 400 from any used skill, to be squatting. Figures 8 and 9 show that a lower value of the threshold benefits from a smaller initial setup time and a lower FRR (i.e., better usability). However, a higher threshold improves the security properties by reducing the instances of invocation confusion. We find that a distance threshold of 400 provides an appropriate trade-off

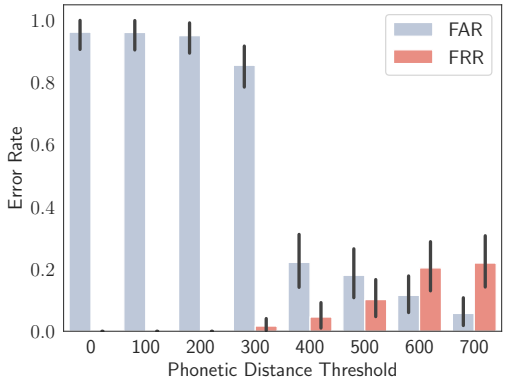


Fig. 8. Error Rate and Security-Usability Tradeoff: describes the error rates (FAR and FRR) for different phonetic graph distance thresholds.

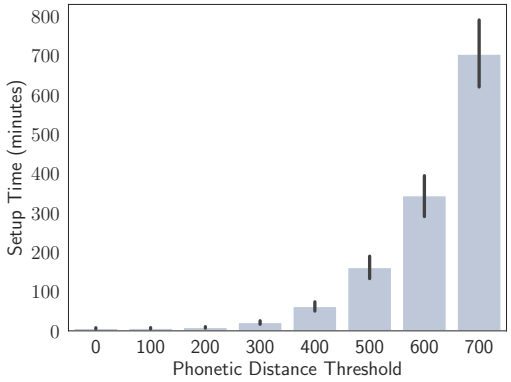
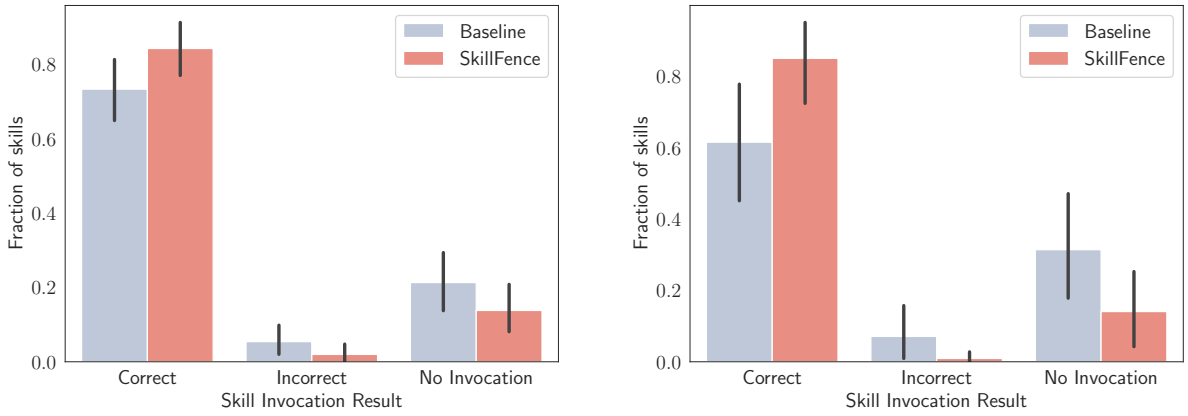


Fig. 9. Initial Setup Time: describes SkillFence's initial setup time for different phonetic graph distance thresholds.

between usability and security. Figure 8 shows that across all users ($N = 116$), for a distance threshold of 400, SkillFence provides an FRR of 9.17% and an FAR of 19.83%. Note that the FAR is not 0% because SkillFence did not enable some of the used skills (and correspondingly disable their phonetic neighbors); these skills either did not have a backlink or their counterpart activity was missing.

5.4.2 Performance Overhead. Out of all the user-facing modules of SkillFence, the enabling/disabling phase takes the longest. The primary reason being that we block our extension's operation for two seconds after each new skill page is loaded to avoid Amazon's robot detection that automatically logs the user out. Thus, it takes 2.5 seconds on average to enable a single skill and 3.08 seconds to disable a skill. Figure 9 shows that for a distance threshold of 400, the average initial setup time for a user is about one hour. This operation can be sped up using parallelization at the risk of Amazon blocking SkillFence. The optimal solution to this problem would be to avoid iteration altogether by using enable and disable functions that allow multiple skills to be selected simultaneously. Should this ability be added, the usability cost for a user's initial defense setup would be greatly reduced. Once the setup is completed, subsequent updates to either the mapper table or the user data only trigger enable/disable of a small number of skills which can be completed in around 2-5 minutes.

5.4.3 Runtime Evaluation. We conclude the trace-based evaluation of SkillFence by performing invocation trials for all of the users' used skills. For each user, we run these trials in two configurations: baseline and after initializing SkillFence. To setup SkillFence, we enable the matched skills and disable their phonetic neighbors within a threshold of 400. Figure 10 shows that SkillFence is able to increase correct invocations and reduce incorrect invocations for both TTS and human reconstructed (LibriSpeech) audio. On an average, SkillFence increases correct invocations from 68% to 86% and reduces incorrect invocations from 6.2% to 1.8% for a real world user — the rest corresponds to no invocation instances. Across all users, SkillFence is able to increase correct invocations from 194 to 251, and reduce incorrect invocations from 23 to 6. Out of the 6 incorrect invocations, 4 were due to the lack of secure backlink and 2 due to lack of counterpart data. *There are no incorrect invocations for skills with secure backlink and counterpart activity.* Additionally, SkillFence reduces the instances where Alexa is unable to invoke any skill.



(a) TTS

(b) LibriSpeech

Fig. 10. Trace based runtime end to end evaluation of SkillFence. Average Fraction of correct, incorrect and unsuccessful invocations per-user in the user study.

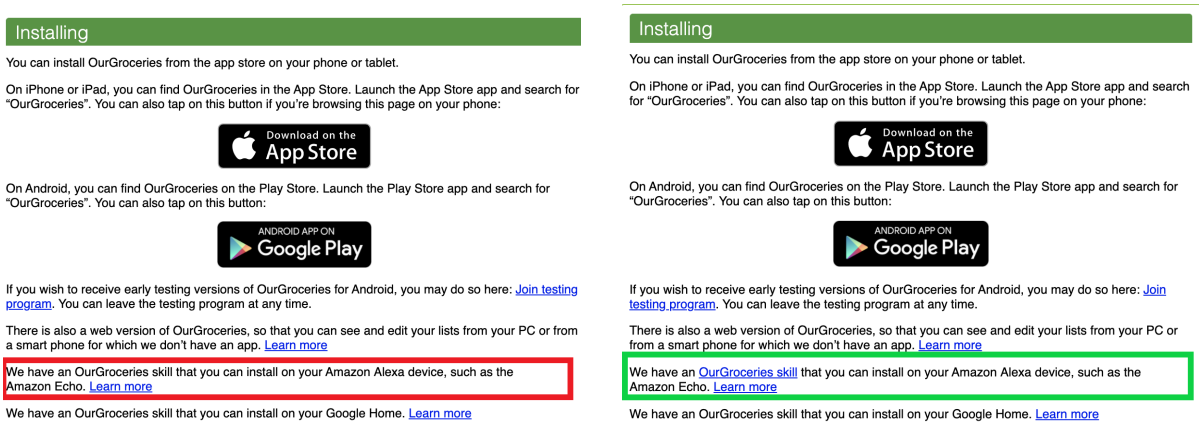
5.5 Q3. Security and Privacy Analysis

Per our threat model, the attacker is a malicious skill that tries to get executed by leveraging various design flaws in the Alexa ecosystem. The user's browser, smartphone and the legitimate skill developer's websites are trusted and not compromised. An attacker could try to compromise a legitimate website to include a fake link to their attack skill. Because of SkillFence however, the bar for the attack is now much higher than just uploading a malicious skill to a loosely-vetted marketplace.

5.5.1 Adaptive Attack Analysis. We consider an adaptive attacker with knowledge of how SkillFence works. Concretely, this attacker can: (1) trick the user into visiting a fake website such that the counterpart activity is poisoned; (2) inject text on a legitimate skill developer website that supports user-generated content so that the backlink analysis is tricked; (3) create a skill whose invocation phrase is just outside the phonetic radius of the legitimate skill in an attempt to still leverage voice confusion attacks. As we discuss below, our work prevents these attacks by design.

Counterpart activity poisoning. An attacker can trick the user into clicking on websites they control and link those websites to a malicious skill. This can result in malicious skills getting enabled because SkillFence mistakenly interprets the malicious websites as the user's intention. SkillFence prevents this attack using its website history filtering technique discussed in Section 4.4.1. An attacker could also try to trick the user into installing an Android app they control. This is a harder task than just uploading a malicious skill to a loosely-vetted marketplace. The attacker has to create an Android app, get it past Google Play's strict vetting policies [6, 12] and then trick the user into installing the application. Thus, SkillFence raises the bar for a voice confusion attack.

Fake URL injection on legitimate website. If a legitimate skill developer's website supports user-generated content (e.g., a help forum), an attacker can insert a URL to their attack skill, thus tricking SkillFence into linking the legitimate website's identity with the fake skill. Our backlink analysis prevents this attack by excluding user-generated pages.



(a) No backlink present on OurGroceries website (before).

(b) OurGroceries added a backlink URL (current).

Fig. 11. The updates to *OurGroceries*' website based on our recommendation.

Phonetic threshold attack. The malicious skill developer can choose an invocation phrase that lies just outside the phonetic cluster of an enabled skill. This can lead to a voice-based confusion attack. In Section 5.3, we demonstrate that the probability of incorrect invocation decreases for a higher threshold. Hence, choosing a conservative threshold makes it harder to induce a voice confusion attack.

5.5.2 Privacy Analysis. The SkillFence extension accesses website history and list of installed Android apps. However, it does not store any of this information, not even locally. It also does not transmit it to the server, and only receives the mapper table. It just computes on this information locally using the mapper table. The only information communicated to Amazon is the set of enabled and disabled skills. The set of enabled ones can leak a small amount of information to Amazon if there are skills that user does not eventually use. Concretely, it leaks information that a website corresponding to that skill was visited. This is a small amount of information compared to existing privacy issues with systems like voice assistants [6, 12, 20] and is reasonable compared to the security advantage of preventing voice confusion attacks.

6 DESIGN RECOMMENDATIONS

Based on our experience in designing SkillFence and evaluating its performance with real user data, we distill the following key recommendations. These recommendations, if implemented by voice assistant manufacturers, broadly limit the impact of voice confusion attacks in the voice assistant ecosystem and improve the efficacy of SkillFence. We observe that these recommendations are simple to implement by the appropriate stakeholders.

6.1 List Skill Marketplace URLs

We recommend that every skill developer update their website to include a URL to the Amazon marketplace (or the corresponding market in case of other voice assistant manufacturers) listing of their skill. Skill developers already include user guides on their websites for the Alexa skills that they develop. Including this link is a simple change. Indeed, there is precedent for this type of linking — it is very common for developers to include links to the Android and iOS apps they build for their services. If all developers did this, then SkillFence will be able to derive secure identity for all skills in the marketplace. Amazon, as a major stakeholder in the space, can require skill developers to host such a link. A related recommendation is that Amazon also updates their skill marketplace

website to include a direct URL to the skill developer's webpage that hosts the backlink. This will remove the need for a backlink search process.

We began outreach efforts in contacting skill developers to implement this change. We especially credit the *OurGroceries* developers for swiftly implementing this change (Figure 11).

6.2 Validate Skill Metadata

We recommend that voice assistant manufacturers like Alexa properly vet the metadata of skills on the marketplace. For example, if a skill developer lists a certain URL for their privacy policy, Amazon should perform domain-ownership testing to ensure that indeed, this skill developer owns the domain where the privacy policy is hosted. This test will limit the ability of attackers to mirror legitimate skills. However, this recommendation represents a larger infrastructural and procedural change to the Alexa ecosystem compared to the first recommendation above.

6.3 Provide Better Skill Invocation Control

We recommend that voice assistant platforms provide an efficient API to control skill invocation. Systems like SkillFence will benefit from an API that can batch-allow a certain set of skills to be eligible for execution while batch-disabling a set of skills to be *ineligible* for execution despite confusing voice commands. Our work extracts this property from Alexa's skill enable/disable API, but its inflexible nature requires optimizations that trade-off performance and security. We also recommend that Amazon clarify the behavior of their Enable and Disable mechanisms. In this work, we empirically verify that the Enable primitive prioritizes the enabled skill over other skills. However, a concrete understanding of both the extent to which an enabled skill is preferred and the effects of enabling multiple skills (phonetically similar or not) cannot be gained without an explanation of how the enable primitive interacts with Alexa's ASR. Furthermore, if the details of the enable/disable mechanisms are published, SkillFence will be able to dispense security recommendations that are globally applicable and deal with every conceivable interaction with these two primitives.

7 LIMITATIONS

7.1 Applicability to Other Voice Assistants

Our design and implementation of SkillFence consider Alexa because it is the most popular voice assistant system. It requires three components: an invocation phrase, a notion of secure identity, and control over which skill executes. These components apply to the Google Voice Assistant, the other popular voice assistant system. First, the Google Voice Assistant, allows users to interact with apps (called actions) using their names - "Let me speak/talk to <app name>". The conflicts from the action name induce uncertainties for this platform as well. Second, the Google voice assistant ecosystem lists the actions and links to developer websites and privacy policies. We verified that a similar notion of secure identity could be achieved using the search and crawl approach.[6] Third, the Google Assistant has a subset of actions that allow users to link their accounts. The users can also unlink a linked action. It is, however, unclear how the link/unlink operation affects the action invocation process — this will require additional experimentation that we leave to future work.

7.2 Low Availability of Secure Backlinks

We were able to find secure backlinks for 474 out of 3659 skills on the Amazon Alexa market. Although the proportion (72 out of 162 skills) was higher for skills actually used by real world users (from the user study), it is still inadequate. However, this number is likely to increase as almost all developers list android/iOS app

[6]Starbucks is an example: www.starbucks.com/promo/googleassistant

links on their websites — it is only a matter of time before they start adding links for their Alexa skills. We are also engaging in outreach efforts by contacting developers and helping them understand the value of adding backlinks to their websites. Upon our recommendation, the developers of OurGroceries (Figure 11) have already updated their website to include a backlink to their Alexa skill.

7.3 Usability Study

One limitation of our evaluation is that we do not directly evaluate users' experiences with SkillFence. Such an analysis would involve tracking user interactions with SkillFence and the Alexa skill ecosystem over an extended period of time. While it would provide useful insight into SkillFence's adoption and continued usage rates, such study is out of scope for this work; in this paper, we lay out SkillFence's motivation, design and effectiveness. Hence, we leave that to future work. However, we do perform an evaluation of SkillFence's usability by relying on standard metrics like the False Acceptance Rate (FAR) and False Rejection Rate (FRR). We also study the overhead incurred by the end-users while using SkillFence in terms of the initial setup time.

7.4 Enable/Disable Setup Time

SkillFence requires an average initial setup time of about one hour. The primary reason for this overhead is that the Alexa skill API only offers a method to enable or disable a single skill at a time. Attempts to parallelize these operations run the risk of Amazon blocking SkillFence. SkillFence tries to address this usability overhead by performing most of the operations in the background. Also, once the setup is completed, later updates can be perform much faster (around 2-5 minutes). We recommend that voice assistant manufacturers provide an API that can efficiently disable a large list of skills with a single network call.

8 CONCLUSION

We propose SkillFence, a browser extension that prevents voice-based confusion attacks on voice assistants like Amazon Alexa. Our system analyzes a user's counterpart activity to reduce dis-ambiguities in the skill invocation process and ensure only the skills preferred by the user execute despite ambiguity. Using real user data, we evaluate the different components of SkillFence and demonstrate its utility in protecting existing users. We distill recommendations for stakeholders that, if adopted, will increase the effectiveness of SkillFence, and we are beginning to see adoption.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their constructive feedback that has made the work stronger. This work was supported in part by the University of Wisconsin-Madison Office of the Vice Chancellor for Research and Graduate Education with funding from the Wisconsin Alumni Research Foundation. This work was also supported in part by DARPA (through the GARD program) and the NSF through awards: CNS-1838733, CNS-1942014, and CNS-2003129.

REFERENCES

[1] N. Carlini, P. Mishra, T. Vaidya, Y. Zhang, M. Sherr, C. Shields, D. Wagner, and W. Zhou. 2016. Hidden Voice Commands. In *25th USENIX Security Symposium (USENIX Security 16)*.

[2] Yuxuan Chen, Xuejing Yuan, Jiangshan Zhang, Yue Zhao, Kai Zhang, Shengzhi Chen, and XiaoFeng Wang. 2020. Devil's Whisper: A General Approach for Physical Adversarial Attacks against Commercial Black-box Speech Recognition Devices. In *29th USENIX Security Symposium (USENIX Security 20)*.

[3] E. Fernandes, J. Paupore, A. Rahmati, D. Simionato, M. Conti, and A. Prakash. 2016. FlowFence: Practical Data Protection for Emerging IoT Application Frameworks. In *Proceedings of the 25th USENIX Security Symposium*.

[4] Dmitry Gerasimenko. 2010 (accessed 2020). *Ahrefs*. https://ahrefs.com

[5] 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. https://www.usenix.org/conference/usenixsecurity20/presentation/guo

[6] Hang Hu, Limin Yang, Shihan Lin, and Gang Wang. 2020. A Case Study of the Security Vetting Process of Smart-home Assistant Applications. In *IEEE Workshop on the Internet of Safe Things (SafeThings)*.

[7] Amazon Inc. [n.d.]. Alexa Skill Certification Guidelines. https://developer.amazon.com/en-US/docs/alexa/custom-skills/certification-requirements-for-custom-skills.html.

[8] Amazon.com Inc. [n.d.]. Alexa Skills for Business and Finance. https://www.amazon.com/Alexa-Skills-Business-Finance/b?ie=UTF8&node=14284819011.

[9] BRET Kinsella. 2018. Should Amazon Alexa Stop Allowing Duplicate Invocation Names? Should Google Assistant Permit Them? https://voicebot.ai/2018/03/26/amazon-alexa-stop-allowing-duplicate-invocation-names-google-assistant-permit/.

[10] 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, Baltimore, MD, 33–47. https://www.usenix.org/conference/usenixsecurity18/presentation/kumar

[11] 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)*. 33–47.

[12] Christopher Lentzsch, Sheel Jayesh Shah, Benjamin Andow, Martin Degeling, Anupam Das, and William Enck. 2021. Hey Alexa, is this Skill Safe?: Taking a Closer Look at the Alexa Skill Ecosystem. In *Proceedings of the 28th ISOC Annual Network and Distributed Systems Symposium (NDSS)*.

[13] Christopher Lentzsch, Sheel Jayesh Shah, Benjamin Andow, Martin Degeling, Anupam Das, and William Enck. 2021. Hey Alexa, is this Skill Safe?: Taking a Closer Look at the Alexa Skill Ecosystem. In *Proceedings of the 28th ISOC Annual Network and Distributed Systems Symposium (NDSS)*.

[14] Yiqun Liu, Rongwei Cen, Min Zhang, Shaoping Ma, and Liyun Ru. 2008. Identifying Web Spam with User Behavior Analysis. In *Proceedings of the 4th International Workshop on Adversarial Information Retrieval on the Web* (Beijing, China) *(AIRWeb '08)*. Association for Computing Machinery, New York, NY, USA, 9–16. https://doi.org/10.1145/1451983.1451986

[15] 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. arXiv:1910.14112 [cs.HC]

[16] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur. 2015. Librispeech: An ASR corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 5206–5210. https://doi.org/10.1109/ICASSP.2015.7178964

[17] Paul Cutsinger. 2018. How to Improve Alexa Skill Discovery with Name-Free Interaction and More. https://developer.amazon.com/blogs/alexa/post/0fecdb38-97c9-48ac-953b-23814a469cfc/skill-discovery.

[18] Ritik Singh. 2021. 7 Ways to Find If an App Is Fake or Real Before Installing It. https://gadgetstouse.com/blog/2021/04/19/find-app-is-fake-or-real-before-installing/.

[19] Nirupam Roy, Sheng Shen, Haitham Hassanieh, and Romit Roy Choudhury. 2018. Inaudible Voice Commands: The Long-Range Attack and Defense. In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*. USENIX Association, Renton, WA, 547–560. https://www.usenix.org/conference/nsdi18/presentation/roy

[20] 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 (WWW'20)*.

[21] Takeshi Sugawara, Benjamin Cyr, Sara Rampazzi, Daniel Genkin, and Kevin Fu. 2020. Light Commands: Laser-Based Audio Injection Attacks on Voice-Controllable Systems. In *29th USENIX Security Symposium (USENIX Security 20)*.

[22] R. M. Suresh and R. Padmajavalli. 2007. An Overview of Data Preprocessing in Data and Web Usage Mining. In *2006 1st International Conference on Digital Information Management*. 193–198. https://doi.org/10.1109/ICDIM.2007.369352

[23] Understand the Smart Home Skill API [n.d.]. Understand the Smart Home Skill API. https://developer.amazon.com/en-US/docs/alexa/smarthome/understand-the-smart-home-skill-api.html.

[24] 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)*.

[25] voicebot.ai. 2021. Alexa Skill Counts Surpass 80K in US, Spain Adds the Most Skills, New Skill Rate Falls Globally. https://voicebot.ai/2021/01/14/alexa-skill-counts-surpass-80k-in-us-spain-adds-the-most-skills-new-skill-introduction-rate-continues-to-fall-across-countries/.

[26] Xuejing Yuan, Yuxuan Chen, Yue Zhao, Yunhui Long, Xiaokang Liu, Kai Chen, Shengzhi Zhang, Heqing Huang, XiaoFeng Wang, and Carl A. Gunter. 2018. Commandersong: A Systematic Approach for Practical Adversarial Voice Recognition. In *Proceedings of the 27th USENIX Conference on Security Symposium* (Baltimore, MD, USA) *(SEC'18)*. USENIX Association, USA, 49–64.

[27] Guoming Zhang, Chen Yan, Xiaoyu Ji, Tianchen Zhang, Taimin Zhang, and Wenyuan Xu. 2017. DolphinAttack: Inaudible Voice Commands. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (Dallas, Texas, USA) *(CCS '17)*. Association for Computing Machinery, New York, NY, USA, 103–117. https://doi.org/10.1145/3133956.3134052

[28] N. Zhang, X. Mi, X. Feng, X. Wang, Y. Tian, and F. Qian. 2019. Dangerous Skills: Understanding and Mitigating Security Risks of Voice-Controlled Third-Party Functions on Virtual Personal Assistant Systems. In *2019 IEEE Symposium on Security and Privacy (SP)*, Vol. 00. 263–278. https://doi.org/10.1109/SP.2019.00016

[29] Yangyong Zhang, Lei Xu, Abner Mendoza, Guangliang Yang, Phakpoom Chinprutthiwong, and Guofei Gu. 2019. Life after Speech Recognition: Fuzzing Semantic Misinterpretation for Voice Assistant Applications. In *Proceedings of the Network and Distributed System Security Symposium (NDSS'19)*.

[30] Yangyong Zhang, Lei Xu, Abner Mendoza, Guangliang Yang, Phakpoom Chinprutthiwong, and Guofei Gu. 2019. Life after Speech Recognition: Fuzzing Semantic Misinterpretation for Voice Assistant Applications.. In *NDSS*.

[31] B. Zhou, S. C. Hui, and A. C. m. Fong. 2006. An Effective Approach for Periodic Web Personalization. In *2006 IEEE/WIC/ACM International Conference on Web Intelligence (WI 2006 Main Conference Proceedings)(WI'06)*. 284–292. https://doi.org/10.1109/WI.2006.36

A ENABLE/DISABLE API EVALUATION SET

Test set for evaluation of skills with identical invocation phrases (Sec. 5.3.1) -

(1) {'Work excuses', 'Work Excuse Generator'} with invocation phrase: 'work excuses'
(2) {'A pat on the Back', 'A pat on the Back'} with invocation phrase: 'a pat on the back'
(3) {'Stock Market', 'Stock Market', 'UPRO Market Price'} with invocation phrase: 'stock market'
(4) {'Joke Master', 'Joke Master', 'Joke Master',} with invocation phrase: 'joke master'
(5) {'Sevenstax Coffee Maker', 'Coffee Maker', 'Coffee Maker', 'Coffee Maker'} with invocation phrase: 'coffee maker'
(6) {'My Home', 'pi home', 'MY HOME', 'My home cake'} with invocation phrase: 'my home'
(7) {'Good Night', 'Sounds: Good Night', 'Good Night', 'Good Night', 'Goodnight With Motivation Success Quotes'} with invocation phrase: 'good night'
(8) {'Inspiring Quotes', 'All Time Inspiring Quotes', 'Inspiring Quotes', 'Inspiring Quotes', 'Inspiring Quotes'} with invocation phrase: 'inspiring quotes'