COMBATING ROBOCALLS TO ENHANCE TRUST IN CONVERGED TELEPHONY

A Dissertation
Presented to
The Academic Faculty

By

Sharbani Pandit

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Computer Science
Department of College of Computing

Georgia Institute of Technology

December 2021

COMBATING ROBOCALLS TO ENHANCE TRUST IN CONVERGED TELEPHONY

Thesis committee:

Prof. Mustaque Ahamad, Advisor
School of Computer Science
Georgia Institute of Technology

Prof. Mostafa Ammar
School of Computer Science
Georgia Institute of Technology

Prof. Roberto Perdisci, Co-Advisor
Computer Science
University of Georgia, Georgia Institute of Technology

Dr. Lillian Lo
Principal Data Scientist
AT&T

Prof. Diyi Yang
School of Interactive Computing
Georgia Institute of Technology

Date approved: Aug 19, 2021

For my partner Krishanu who has been my rock throughout this journey.

ACKNOWLEDGMENTS

Research is a teamwork and is only possible after the time and effort put together by a number of people. I was fortunate enough to have a group of incredibly helpful people in my personal and professional life. A big credit of completing this research goes to these people.

First and foremost I would like to thank my advisor Mustaque Ahamad, who has been a constant source of support and guidance. He has always encouraged me to be a better researcher and never give up when faced with challenges. My co-advisor Roberto Perdisci has taught me to learn from my mistakes and always pushed me further to achieve what might have seemed unattainable. The quality of the work in the thesis would have been lower without his help. I was fortunate enough to learn from two incredible advisors and I express my earnest gratitude and appreciation to you.

I would also like to thank Diyi Yang, collaborator and PhD committee member who had no small part to play in the formation of the idea and who always had valuable advice on how to improve the work in this dissertation. Also, many thanks to the other PhD committee members Mostafa Ammar and Lillian Lo for serving on my committee and for their valuable feedback and guidance throughout this process.

The work in this dissertation would be been incomplete without access to real world data. It was critical in building the models and conducting the evaluations discussed in this dissertation. Pindrop and Nomorobo provided us with real world data. A special thanks goes to Payas Gupta from Pindrop for being a patient mentor especially during the initial years. I would also like to thank Aaron from Nomorobo for his involvement in the research discussed in this dissertation.

I am grateful for the support I received from my labmates, peers and colleagues. I would especially like to thank Jienan Liu, Yiwen Zhou, Irfan Ozen, Karthika and Ravi. I am also grateful to the Institute of Security and Privacy (IISP) support staff, Elizabeth Ndongi and

Trinh Doan for being ever helpful and taking care of all the administrative paperwork.

Graduate life is hard and it becomes even more difficult when you are trying to adjust in a completely new country and new culture. I want to thank my friends Anita, Anup, Jyoti, Shafin, Antora, Johana and the Bangladeshi Students' Association for being my emotional support and making this transition smooth for me.

I am forever indebted to my family for the love and support. Many thanks to my grandmother, uncles and in-laws for always showering me with their love and blessings. I want to especially thank my mother who raised me single-handedly in a hostile environment. In spite of all the challenges she faced, she never gave up on my dreams and inspired me to reach for the stars. I would not have been where I am without you.

PhD is a long and challenging journey and I want to thank my husband Krishanu for being the shoulder to cry on throughout this journey. He was always there pick up the pieces and encouraged to move forward during my frustration spells. He is the biggest cheerleader of my achievements and I would not have been able to complete this journey without his love and support. I thank you from the bottom of my heart.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Telephone scams are now on the rise and without effective countermeasures there is no stopping. The number of scam/spam calls people receive is increasing every day. YouMail estimates that June 2021 saw 4.4 billion robocalls in the United States and the Federal Trade Commission (FTC) phone complaint portal receives millions of complaints about such fraudulent and unwanted calls each year. Voice scams have become such a serious problem that people often no longer pick up calls from unknown callers. In several scams that have been reported widely, the telephony channel is either directly used to reach potential victims or as a way to monetize scams that are advertised online, as in the case of tech support scams. The vision of this research is to bring trust back to the telephony channel. We believe this can be done by stopping unwanted and fraud calls and leveraging smartphones to offer a novel interaction model that can help enhance the trust in voice interactions. Thus, our research explores defenses against unwanted calls that include blacklisting of known fraudulent callers, detecting robocalls in presence of caller ID spoofing and proposing a novel virtual assistant that can stop more sophisticated robocalls without user intervention.

We first explore phone blacklists to stop unwanted calls based on the caller ID received when a call arrives. We study how to automatically build blacklists from multiple data sources and evaluate the effectiveness of such blacklists in stopping current robocalls. We also used insights gained from this process to increase detection of more sophisticated robocalls and improve the robustness of our defense system against malicious callers who can use techniques like caller ID spoofing.

To address the threat model where caller ID is spoofed, we introduce the notion of a virtual assistant. To this end, we developed a Smartphone based app named RobocallGuard which can pick up calls from unknown callers on behalf of the user and detect and filter out unwanted calls. We conduct a user study that shows that users are comfortable with a virtual

assistant stopping unwanted calls on their behalf. Moreover, most users reported that such a virtual assistant is beneficial to them. Finally, we expand our threat model and introduce RobocallGuardPlus which can effectively block targeted robocalls. RobocallGuardPlus also picks up calls from unknown callers on behalf of the callee and engages in a natural conversation with the caller. RobocallGuardPlus uses a combination of NLP based machine learning models to determine if the caller is a human or a robocaller. To the best of our knowledge, we are the first to develop such a defense system that can interact with the caller and detect robocalls where robocallers utilize caller ID spoofing and voice activity detection to bypass the defense mechanism. Security analysis explored by us shows that such a system is capable of stopping more sophisticated robocallers that might emerge in the near future. By making these contributions, we believe we can bring trust back to the telephony channel and provide a better call experience for everyone.

CHAPTER 1
INTRODUCTION

1.1 Threats to Trust in Telephony

Telephone has been a trusted voice communication medium for over 140 years. Currently, about 4.77 billion people have access to the telephony system. In addition to personal communication, important business transactions are enabled by the phone system. Recent technological advances and changes in communication policies offer many benefits; including lower cost and richer interactions. However, with its convergence with the Internet, cyber criminals are now using the telephony channel to craft new attacks [1]. Robocalling, voice phishing [2] and caller-id spoofing [3] are some of the techniques that are used by fraudsters and criminals in these attacks. The number of scam/spam calls people receive are increasing every day. YouMail estimates that June 2021 saw 4.4 billion robocalls [4] in the United States and the Federal Trade Commission (FTC) phone complaint portal receives millions of complaints about such fraudulent and unwanted calls each year [5]. In several scams that have been reported widely, the telephony channel is either directly used to reach potential victims or as a way to monetize scams that are advertised online, as in the case of tech support scams [6].

A survey of telecom service providers in 2015 estimates the losses due to fraud to 38.1 billion US dollars. This constitutes 1.69% of the estimated global revenue [7]. In addition to the financial losses, fraud aiming at service disruption or reputation damage may have devastating effects, because the telecommunications network is a critical infrastructure with millions of users relying on it. Perpetrating fraud in telecom networks is relatively easy. Most of the attacks can be performed remotely and they do not require costly equipment or high level of technical expertise. Moreover, it is often very easy to obtain a financial

benefit from telephony fraud [8]. Often, fraud is buried in massive volume of traffic and large variety of services. Therefore, it is difficult to identify, detect and prevent.

1.1.1 Robocalls, Voice Spam and Caller ID Spoofing

Voice spam is one of the most visible types of voice fraud targeting customers. It includes all types of unsolicited and illegitimate calls. Fraudsters can obtain phone number lists from leaked databases, form submissions, or simply by purchasing them online [9]. They mostly use autodialers to generate large number of calls and use prerecorded messages (robocalling) which may be later forwarded to live call center agents to interact with the victims. Caller ID spoofing and social engineering techniques are frequently used to deceive people to perform certain actions or to reveal sensitive information. Due to the low cost and scalability of VoIP based calling systems, scammers can make millions of calls and easily expand the scam ecosystem. Voice spam can take many forms, here we explain some of the most common schemes. Telemarketing is a method of direct marketing in which a salesperson entices customers to buy products or services over the phone [17]. Telemarketing can be illegitimate in certain jurisdiction, e.g., if the telemarketer did not take prior consent from the call recipient [10]. In voice phishing (also known as vishing), the caller impersonates a legitimate organization, person or entity and tries to gain access to private, personal and financial information using social engineering [11, 12].

Caller ID spoofing is often used by scammers to hide their real identity and makes it difficult to block the spam calls or to take legal actions against fraudsters [13]. Many other types of scams can make use of telephony. For example, in the tech support scam, fraudsters try to convince people that their computer is infected with malware (mostly by tricking them into installing remote access tools) and request a payment to solve the so-called problem [14]. In advance fee fraud (419 scam), the victim is being tricked into making some up-front payment to be able to receive a larger sum of money, such as a bogus lottery prize [15]. A similar scam is the free cruise scam, where fraudsters advertise

a free cruise opportunity, but later on require additional payments [16]. Such automated robocalls are used to scale attacks at low cost. Moreover, we expect robocallers to get more sophisticated in the near future to evade measures designed to stop them. In this dissertation we focus on detecting and preventing current and future threats posed by unwanted callers via the telephony channel.

1.2 Countering Telephony Threats

At a high level, the robocall problem resembles the email spam problem, in which information about the source of an email could potentially be spoofed. Over the years, the security community has been successful in developing effective spam filtering solutions to ensure that email remains a viable means of communication [17, 18]. However, most techniques used in such solutions cannot be applied to detect and filter voice spam. This is because when a user receives a phone call, the only information the user can rely on before answering is the caller ID (i.e., the calling phone number). Namely, the context of the call (i.e., the content of the caller's message) cannot be used to detect if the call is spam or not. In absence of such context, building blacklists using metadata such as caller ID, historical caller behavior is the only viable way to stop malicious callers. Blacklists have been investigated for domain names, IP addresses and other online resources such as URLs to combat email spam and malware infections [19, 20]. Similar to domain or IP blacklisting, phone blacklists have emerged which use the caller ID to decide whether to block a call or not. However, domain or IP blacklists typically utilize network, email content and other application specific features to block such malicious entities which differ significantly from information available in phone abuse datasets.

In response to the increasing number of unwanted or fraudulent phone calls, a number of call blocking applications have appeared on smartphone app stores, some of which are used by hundreds of millions of users (e.g., [21, 22, 23]). Additionally, a recent update to the default Android phone app alerts users of suspected spam calls [24]. However, little

is known about the methods used by these apps to identify malicious numbers and how accurate and effective these methods are in practice.

1.2.1 Telephony Blacklists

To address telephony threats, in this dissertation, we first look at detection of calls from unwanted sources and their blocking via blacklisting, which has been somewhat effective against online abuse. We systematically investigate multiple data sources that may be leveraged to automatically learn phone blacklists, and explore the potential effectiveness of such blacklists by measuring their ability to block future unwanted phone calls. Specifically, we consider four different data sources: user-reported call complaints submitted to the Federal Trade Commission (FTC) [25], complaints collected via crowd-sourced efforts, such as 800notes.com and MrNumber.com, call detail records (CDR) from a telephony honeypot [26] and honeypot-based phone call audio recordings. We provide a detailed analysis of how such data sources could be used to automatically learn phone blacklists, measure the extent to which these different data sources overlap, explore the utility of call context for identifying phone spam campaigns, and evaluate the effectiveness of these blacklists in terms of unwanted call blocking rates.

Because the utility of phone blacklists comes from blocking calls which are based on the calling phone number, another important challenge is presented by caller ID spoofing. Such spoofing is easy to achieve, and robocallers have resorted to tricks like neighbor spoofing (caller ID is similar to the targeted phone number) [27] to overcome call blocking and to increase the likelihood that the targeted user will pick up the call. Hence, call blocking techniques based on phone blacklists can be somewhat effective against spam calls; however, their effectiveness can be easily degraded with caller ID spoofing. To help reduce caller ID spoofing, both industry groups and regulatory bodies have explored stronger authentication for call sources. Although the recently published IETF RFC 8588 describes SHAKEN/STIR approach to enhance trust in the source of a call with signatures [28, 29,

30], its widespread deployment by carriers will likely take many years. Furthermore, elimination of caller ID spoofing will not make all unwanted calls go away, as phone numbers can be cheaply acquired and used to overcome blacklists.

1.2.2 Engaging Callers to Detect Robocalls

To detect robocalls when caller ID can be spoofed and to provide the user with more meaningful call context, we propose ***RobocallGuard***, a natural voice interaction model which is mediated by a Virtual Assistant(VA). The VA mimics a human call screener (e.g., a secretary) who picks up an incoming phone call and makes the user aware of the call only when it confirms that the call is not a robocall or other type of spam. When a call arrives, if the caller ID is not among the user's contact list, the VA transparently picks up the call and briefly interacts with the caller to determine if its source is a robocaller. Such interaction aims to be natural for legitimate callers, while enabling the detection of robocall sources who indiscriminately target a large number of victims. Furthermore, such interaction with the VA enables learning the context of the call. Calls that are not detected as spam are passed on to the user, and the context extracted from the conversation between the VA and the caller is provided simultaneously, allowing the user to make an informed decision on whether the call is unwanted or legitimate.

Recently, a number of automated caller engagement systems that attempt to collect information about a call source have been proposed. The Call Screen feature available on the latest versions of the Android phone app provides call context to the user via a real time transcript of the call audio. When an incoming call arrives, the user is prompted with three options: answer, decline and screen. If the screen option is chosen, Google Assistant engages with the caller to collect audio and generate a transcript of the ongoing call. However, users are notified (i.e., the phone rings) of all incoming calls (including robocalls) and user intervention is needed to screen such calls. In the latest version of the Phone app, Google allows an automatic call screen feature where users can opt to have

their call from unknown callers automatically screened, thus enabling the elimination of user intervention if the user chooses to do so. This feature also claims to block robocalls on behalf of the user. Upon picking up the call, Google Assistant screens the call and asks who's calling and why. Call Screen then detects spam based on what a caller says, or if the call source matches a phone number in Google's database of known spammers and robocallers (when the "Caller and spam ID" setting is on). A detected spam call is then declined without alerting the user. However, once adopted by a large number of users, Call Screen can be easily evaded by future robocallers who need to bypass Google Assistant and reach their targets. Instead of playing the spam content, robocallers can simply play something benign when asked about the purpose of the call, thus evading detention by Google Assistant. Once Google Assistant forwards the call to the callee, robocallers can then expose the spam content to their target.

Robokiller [31] is another smartphone application featuring an Answer Bot that detects spam calls by forwarding all incoming calls to a server, which accepts each call and analyzes its audio to determine if the audio source is a recording. Once the call is determined to come from a human, it is forwarded back to the user. In Robokiller, a caller continues to hear rings while the call is picked up, analyzed and forwarded back to the user, which could negatively impact legitimate callers. Also, the audio analysis techniques used by Robokiller can be countered by a more sophisticated robocaller, and unwanted calls originating from human callers, such as telemarketers or human callers hired by scam campaigns like Tech support, IRS etc., cannot be stopped. To the best of our knowledge, no systematic usability and effectiveness studies have been reported of either Robokiller or Google's Call Screen. Our goal is to explore an automated voice-based interaction approach that maintains both caller and callee user experience, eliminates user interruption and stops unwanted calls from current and future robocallers even in the presence of spoofed calls.

RobocallGuard is the first version of our system to mainly test the usability of a VA and its effectiveness in stopping current robocalls. The survey responses from the user study

we conducted show that both callers and callees are comfortable with the change in the call experience due to the VA and found the VA beneficial. Moreover, [32] has shown that users need an app which handles spam calls without making the phone inoperable. The fact that RobocallGuard filters out spam call without user intervention and without making the phone inoperable adds desired convenience for users.

Much of the current voice abuse over telephony is perpetrated by mass robocallers who indiscriminately call a large number of potential victims. They use techniques like neighbor spoofing to increase the likelihood that their calls are picked up. Our results show that RobocallGuard can be effective against such mass robocallers. However, robocallers might become more sophisticated once systems like RobocallGuard are deployed. Therefore, next, we explore challenges associated with detection of more sophisticated attacks that will target voice interactions in the future.

1.2.3 Combating Future Telephony Abuse

We introduce ***RobocallGuardPlus***, an intelligent virtual assistant capable of detecting sophisticated robocallers. Similar to RobocallGuard, RobocallGuardPlus picks up incoming calls from unknown callers and initiates a conversation with the caller. However, RobocallGuardPlus asks a variety of questions that occur naturally in human conversations and is capable of assessing if a response provided by the caller to a particular question is appropriate or not. To do this, we develop Deep Neural Network (DNN) based models using natural language processing tools. We are faced here with the challenge of deciding the right trade-off between usability and robustness. To ensure usability RobocallGuardPlus when interacting with the caller asks questions that naturally occur in a typical phone conversation. To ensure robustness RobocallGuardPlus asks questions in a random order and makes sure that no specific pattern exists which can be exploited by malicious actors. To evaluate usability we conducted an IRB approved user study and demonstrated that user experience is preserved by RobocallGuardPlus. Our red team based security analysis shows

that RobocallGuardPlus improves the robustness of our defense system and can identify sophisticated robocallers who do more than mass robocalling.

Stopping robocalls requires a defense-in-depth approach and with each step of our research, we handle a more sophisticated threat model and improve the robustness of our defense. We believe each step is a building block working towards our end goal of a more trusted telephony channel.

1.3 Dissertation Hypothesis

The goal of this research is to study robocalls that deliver voice spam and explore methods to combat them to bring trust back to the telephony channel. We believe this can be done by stopping unwanted and fraud calls and offering a natural, novel phone call interaction model that can help enhance the trust and effectiveness of voice interactions. Thus, our research looks at blacklisting, explores defenses against caller ID spoofing and detection of more sophisticated attacks that could come over the voice channel. *Our hypothesis is that by interposing an automated agent between the caller and the callee, a significant number of robocalls can be stopped without interrupting the user while delivering legitimate calls to the user.* To support the hypothesis, this proposal makes the contributions outlined below.

1.4 Contributions

We make the following contributions that enhance our understanding of ways to combat voice spams. We start with a defense that requires no interaction with the caller and blocks calls based on caller ID. We then introduce an interactive agent that makes a natural conversation with the caller to handle more sophisticated threats.

- For no caller interaction defense we present the first systematic study to evaluate the effectiveness of phone blacklists. To understand the nature and effectiveness of phone blacklists, we make the following contributions.

  - We first analyze the characteristics of multiple data sources that may be leveraged to automatically learn phone blacklists, and then measure their ability to block future unwanted phone calls.
  - We investigate a number of alternative approaches for building phone blacklists. In particular, we propose methods for learning a blacklist when call context (e.g., complaint description or phone call transcripts) is available, and when context is missing.
  - We evaluate the effectiveness of the phone blacklists we are able to learn, and show that they are capable of blocking a significant fraction of future unwanted calls (e.g., more than 55% of unsolicited calls). Also, they have a very low false positive rate of only 0.01% for phone numbers of legitimate businesses.
  - To link phone numbers that are part of long running spam campaigns, we apply a combination of unsupervised learning techniques on both user-reported complaints as well as from phone call audio recordings. We then identify the top campaigns from each data source, and show how effective blacklists could be as a defense against such campaigns.

- We explore an automated voice-based interaction approach that seeks to maintain both caller and callee user experience, eliminates user interruption and stops unwanted calls even in the presence of caller ID spoofing. Although it may not be possible to stop all unwanted calls, we believe more trusted communication via the telephony channel can be supported by an automated call screening agent that can detect and block such calls without degrading user experience.
  - To the best of our knowledge, we are the first to evaluate a call screening virtual assistant that uses automated call handling and audio analysis to defend against robocalls and other types of spam calls, including those that evade blacklists with caller ID spoofing. In addition, transcription of call audio recorded by the

virtual assistant is used to provide meaningful context about incoming calls to a user when the phone rings.

  - Our virtual assistant aims to provide a mechanism that is similar, albeit much less "sophisticated", to having a human call screener who can pick up phone calls and only forward to the user those calls which are likely wanted and record messages for the calls that are most likely unwanted. As a result, users are not annoyed with continuous ringing from unwanted calls.
  - To demonstrate the ability of the virtual assistant to detect robocalls, we developed a proof-of-concept smartphone app named RobocallGuard. To this end, we experimented with a corpus of 8,000 real robocalls collected by a large phone honeypot, and show that all of them can be detected and thus blocked.
  - In addition, our proof-of-concept app allowed us to conduct an institutional review board (IRB)-approved user study to assess the usability of our virtual assistant. The results of this study demonstrate that the natural experience of a typical phone call is preserved for both callers and receivers, while benefiting from the ability to detect robocalls and other potentially unwanted calls.

- To handle an expanded threat model that includes targeted attacks, we augment our virtual assistant by introducing a voice interaction enabled smart agent which can initiate a conversation with the caller and can successfully identify more sophisticated robocallers.

  - We design a smart, interactive virtual assistant (RobocallGuardPlus) which can pick up and screen phone calls on behalf of the user. It aims to make a natural conversation with the caller and requires human-like interaction from the caller for the call to be brought to the attention of the callee.
  - RobocallGuardPlus uses a combination of NLP based machine learning models to determine if the caller is a human or a robocaller. To the best of our knowl-

edge, we are the first to develop such a defense system that can interact with the caller and detect robocalls where robocallers utilize caller ID spoofing and voice activity detection to bypass the defense mechanism.

– To demonstrate the ability of the virtual assistant to detect robocalls, we have developed a proof-of-concept defense system named RobocallGuardPlus. To this end, we experimented with a corpus of 8,000 real robocalls collected by a large phone honeypot, and showed that 95% of the mass robocalls, 82% of the evasive robocalls and 75% of the targeted robocalls can be detected and thus blocked.

1.5 Dissertation Outline

This rest of this dissertation is organized as follows. In Chapter 2, we present the related work and explain how our contributions can advance the state-of-the-art.

In Chapter 3 we discuss our work on developing phone blacklists. In Section 3.2 we describe our data collection process in detail. We present the multiple data sources that can be leveraged to build phone blacklists. We also describe a number of high-level insights that can be gained from the multiple data sources. In Section 3.3 we demonstrate how phone blacklists can be learned both when the context of a call is known and when such context is not available.

In chapter 4 we present the evaluation of the phone blacklists we constructed in Section section 3.3. We define our performance metrics and discuss our experimental setup in Sections 4.2 and 4.3. We first show that our blacklists are representative of real-world phone blacklisting applications, by vetting them against two commercial third-party telephony security services in Section 4.4. In addition, we perform an analysis on how well the blacklists we construct can help to block future spam phone calls in Sections 4.5 and 4.6. We discuss the measured false positive rate and our findings on popular spam campaigns in Sections 4.7 and 4.8. In Section 4.9 we cover the limitations of the system. Lastly in

Section 4.10 we conclude our work on evaluation of phone blacklists.

In Chapter 5 we introduce RobocallGuard, a novel voice interaction model which is mediated by a Virtual Assistant(VA). In Section 5.2 we describe our system design in detail. We present our design goals and discuss out threat model. We further describe the virtual assistant workflow and system architecture. Section 5.3 depicts our implementation choices and motivation to implement RobocallGuard as a VoIP application. In Section 5.4, we report the results of experiments we conducted to measure the accuracy of decisions made by our VA for incoming calls, and discuss a user study that was conducted to evaluate the usability of our prototype. Section 5.5 presents the discussion and limitations of our system. Finally Section 5.6 concludes the work on RobocallGuard and sheds light on the feasibility of such voice assistant systems to block robocalls in the future.

In Chapter 6 we introduce RobocallGuardPlus that can handle an expanded threat model. In Section 6.2 we describe our system design in detail. We discuss the threat model, the workflow and system architecture of RobocallGuardPlus. We describe the design and motivation behind each component. In Section 6.3, we report the results of experiments conducted to evaluate our system. The IRB approved user study is described in detail and a red team based security analysis is presented. We discuss the limitations and future work in Section 6.4. Finally, in Section 6.5 we conclude the work.

Chapter 7 concludes the dissertation. In Section 7.1 we summarize the contributions and then in Section 7.2 we discuss limitations of our work. Section 7.3 presents future work. Lastly, Section 7.4 concludes with closing remarks.

CHAPTER 2
RELATED WORK

There has been considerable amount of past research on understanding and combating abuse in the phone channel. We discuss both past work and put the contributions of this dissertation in the context of these works, highlighting the novelty of these contributions. This chapter discusses related work that is relevant to the research explored in this dissertation.

2.1 Telephone Spam Ecosystem

Abuse in the telephony channel has grown considerably and high volume scams that rely on this channel have proliferated in recent years [9, 13]. Recent increase in attacks over the telephony channel can be attributed to the availability of IP telephony (Voice over Internet Protocol). Such calls can be made at no or low cost at scale in an automated fashion similar to email spam, and criminals are already exploiting the telephony channel to craft attacks such as voice phishing(vishing). Vishing (voice phishing) is a common example of voice abuse where a fraudster exploits the phone channel with voice-based interactions to social engineer victims into scams. Maggie et al. [33] conduct one of the first analysis of modern phone frauds relying on vishing. They analyze the content of the conversations in vishing scams, the geography of the target victims, and the role of automation in vishing scams.

A very common way of disseminating telephone spam is robocalling, which uses an autodialer that automatically dials and delivers a prerecorded message to a list of phone numbers. Security researchers have explored such abuse and specific scams. For example, Costin et. al. [34] investigated how phone numbers are used in various scams based on the analysis of crowd sourced web listings. The tech support scam has been one of the most prominent and have received attention from regulators and law enforcement. This scam makes use of online abuse techniques like malicious advertisements and search poisoning

to get victims to call phone numbers controlled by the scammers. Insights into the online and phone infrastructure used by tech support scams and their tactics have been explored in [6, 35]. The targeting of international students by the IRS scam in the United States was explored in [36]. However these works were specific to certain scams only.

Phoneypot [26] demonstrated the feasibility of using a telephony honeypot to augment abuse information available in existing crowd sourced and self-reported datasets like 800notes and the FTC complaint database. Although the data collected from the honeypot was analyzed to discover various abuse patterns (e.g., debt collection calls), stopping robocalls was not the goal of this work. Telephone spam has increased significantly, defrauding consumers of billions of dollars. Therefore, an effective telephone spam defense is critical. In the following section we describe the related work aimed at detecting spam calls.

2.2 Detection of Spam Calls

In response to the increasing number of unwanted or fraudulent phone calls, a number of call blocking applications have appeared on smartphone app stores, some of which are used by hundreds of millions of users (e.g., Truecaller [22], Youmail [23], Hiya [21], Nomorobo [37] etc.). Most of these commercial applications rely on a blacklist which is built on historical data. Use of caller behavior and call patterns can also be seen in defense against unwanted calls. An analysis of a large dataset of VoIP CDRs is presented in [38]. The authors use unsupervised techniques and analyze different call patterns and utilizes the call patterns to group callers. However, the features used here suffice to group users, but they are not designed to differentiate between spammers and the legitimate callers. Clustering of transcripts recorded by a phone honeypot to identify and block calls from major bad actors was explored in [39] . However, since transcripts is the only abuse information source here, it can only block calls from the campaigns that are seen at the honeypot. Profiling of callers has been investigated by several researchers [40, 41, 42]. However, they assume access to large CDR datasets which have associated privacy concerns and tele-

phony service providers do not make such datasets available due to privacy reasons. Yardi et al. [43] characterized behavioral patterns that disambiguate spammers from legitimate users. Call duration and social network connections are used to separate legitimate callers from spam/scam callers in [44] by developing a global reputation based system for callers. Although low reputation phone numbers can be placed on a blacklist, call duration information and social connections of phone users are not available. Furthermore, blacklists and their evaluation in not addressed by this work.

Since robocalling is widely recognized as a serious problem, it has received significant attention from industry, regulatory bodies and standards bodies. As mentioned earlier, commercially available apps provide defenses against robocalls. Telephone carriers such as AT&T [45], Verizon [46] etc, are also providing users with "call protect" options which can screen incoming phone calls. These commercially available solutions generally use metadata such as source phone number and use blacklists of known fraudulent robocallers to block unwanted calls and alert users about them. Blacklists rely on historical data such as user complaints or honeypot generated information [47, 48]. Although it has been shown that such blacklisting methods can be somewhat effective, we demonstrate that their effectiveness degrades as caller ID spoofing increases [49]. Such spoofing has significantly increased in the recent times (robocall blocking company Hiya reported that 56.7% scams reported by their users relied on neighbor spoofing) [50]. Hence, solutions that rely only on call source metadata in conjunction with blacklists alone cannot address the unwanted robocall problem. One approach to better inform users about the sources of their calls is to limit or prevent caller ID spoofing, which can also improve the efficacy of blacklists.

2.2.1 Caller ID Spoofing

A number of research papers have explored how caller ID spoofing can be detected [3, 51, 52, 53]. For example, if the used signaling protocol can check the state of the calling party when a call is received, it must be busy as the caller is in middle of setting up a call.

If the caller is not busy, this is an indicator of spoofing. There are commercial systems that claim to use such approaches to detect spoofed calls (e.g., TrustID [54]). However, the applicability of these approaches for protecting users from fraudulent calls has not been demonstrated. Also, there is lack of rigorous analysis of effectiveness and usability of commercially available applications. Other systems have explored analysis of the call audio to detect provenance of a call [55]. However, this requires that the call be accepted so audio can be collected for analysis.

In an effort to reduce caller ID spoofing, both industry groups and regulatory bodies have explored stronger authentication for call sources. The Federal Communications Commission (FCC) has asked all telecom companies to start using SHAKEN/STIR by the mid 2021. SHAKEN/STIR [30, 5] describes how a calling party's telecom provider can attach a digital certificate to the call message so that the callee can verify that the caller is who the caller ID says it is. Comcast and AT&T were the first ones to demonstrate a prototype of SHAKEN/STIR that could authenticate call sources in a cross-carrier setting in March 2019, and other carriers like Verizon have announced plans to adopt these protocols. While this an important first step, the robocall crisis will not be completely addressed. Since an authenticated caller ID can only help if the calling and receiving carriers both support SHAKEN/STIR, the full potential of this technology will not be realized until all carriers embrace it. Moreover, it will take time for small and medium sized providers to adopt this technology.

Caller ID authentication will likely reduce the number of robocalls but scammers will still be able to use authenticated phone numbers for malicious purpose before they are detected and calls from them can be blocked. The implementation of STIR/SHAKEN will make it easier to track the reputation of a given phone number, but both the FCC and industry experts emphasize that the change will also inevitably spur criminal innovation in robocalling to evade or manipulate the new implementations. Besides, currently not enough information is known about the entire robocall ecosystem. For example, it is not

fully understood how scammers acquire phone numbers, who the scammers are, where the calls are coming from, how scammers are making money, and who is falling victim to these scams. The effectiveness of caller ID authentication efforts cannot be evaluated without a better understanding of the practices of the robocalling industry and their abuse by malicious actors. For instance, if most of the scammers are operating outside of US, increasing legislative penalties might not be as effective. Moreover, authenticated caller ID might not be as effective if victims continue to fall for scams without caller ID spoofing. Hence, it is unlikely that broader adoption of SHAKEN/STIR will offer a cure-all solution [56].

2.3 Spam Call Detection Challenges

There are several challenges in combating telephone spam that are significantly different from email spam. Unlike email, which can be queued for later analysis, a phone call has an immediacy constraint. A telephone call request is immediate and therefore must be analyzed as soon as it appears, and the defense system must complete analysis and take action within a short window of time to reduce the delay. If a solution adds too much delay to a call request, the legitimate caller may assume that the recipient could not answer the phone and hang up. In addition, the content of a voice call is an audio stream as opposed to the text of an email. Besides, the content of a voice call is only revealed when the call is answered. In contrast, an email spam detection system can easily analyze the content of an email. Moreover, the bar for user acceptance of a voice spam detection system is much higher compared to email. Consumers, rightly, have a very low tolerance for false positives of blocked calls. Phone calls tend to be more urgent and important compared to email, and once a phone call is wrongfully blocked it could have severe consequences. Voice spam detection is especially challenging because of lack of global enforcement. In the United States, a number of laws and regulation exist at both the federal and state levels, such as making robocalling illegal [57], making caller ID spoofing illegal [58], and the

establishment of a national Do-Not-Call Registry [59]. Despite efforts by the US government, robocalling and caller ID spoofing still remains an unsolved problem. Technology and globalization have resulted in telephony networks shifting from a national ecosystem to a global ecosystem. With the use of VoIP service, a telephone spammer can cheaply distribute outbound calls from an overseas location. Because the spammers lie beyond the jurisdiction of US law enforcement authorities, it is hard for law enforcement to prosecute those spammers for breaking the law.

2.4 Audio Captchas to Detect Spam

The virtual assistant systems proposed by us in this dissertation pose a challenge to the callers which they must pass in order for their call to be forwarded. Audio captchas aim at achieving a similar goal where users need to prove that they are not robots. Audio captchas were initially created to enable people who are visually impaired to register or make use of a service that requires solving a CAPTCHA [60]. Typically they consist of a series of spoken words/numbers and some form of audio distortion or background noise [61]. Google, Apple, Microsoft, IBM all have their own audio captchas deployed commercially in their services. However several researchers have explored attacks that can easily break audio captchas [62, 63, 64, 65]. Solanki et. al [66] have demonstrated how off-the-shelf speech recognition systems can be used to break all commercially deployed audio captchas. Hence, currently used audio captchas can not be effective when used against sophisticated robocallers. Fanelle et. al [67] explored cognitive audio captchas where users have to solve puzzles, answer math questions or identify sounds to pass the challenge. However, this disrupts the natural call experience and the security implications of such captchas have not been studied. The challenges our virtual assistants (RobocallGuard and RobocallGuardPlus) pose can be thought of as a Natural Language Processing (NLP) based captcha, where the natural flow of the conversation is maintained to preserve usability. Moreover to pass the challenges posed by RobocallGuardPlus, robocallers need more advanced AI capabili-

ties to truly comprehend what the virtual assistant is asking them to do.

2.5 Dialog Systems and Chatbots

In this section we describe the related works on conversational agents, or dialogue systems. The virtual assistant systems proposed by us are conversational agents that filter out unwanted calls on the user's behalf. Typically dialog systems communicate with users in natural language (text, speech, or both), and fall into two classes: task-oriented [68] and chatbots. Task-oriented dialogue agents use conversation with users to help complete tasks. Dialogue agents in digital assistants (Siri, Alexa, Google Now/Home, Cortana, etc.), give directions, control appliances, find restaurants, or make calls. By contrast, chatbots [69] are systems designed for extended conversations, set up to mimic the unstructured conversations or 'chats' characteristic of human-human interaction, mainly for entertainment, but also for practical purposes like making task-oriented agents more natural [70]. Task based dialog agent has the goal of helping a user solve some task like making an airplane reservation or buying product (e.g. Alexa, Siri etc.). Most commercial dialogue systems use the GUS [71] or frame-based architecture [72], in which the designer specifies frames consisting of slots that the system must fill by asking the user. On the other hand, Chatbots are conversational agents designed to mimic the appearance of informal human conversation. Rule-based chatbots like ELIZA [73] and its modern descendants use rules to map user sentences into system responses. Corpus based chatbots [74, 75] mine logs of human conversation to learn to automatically map user sentences into system responses. These conversational agent systems are enormously data-intensive; Serban et al.[76] estimate that training modern chatbots require hundreds of millions or even billions of words. However, data on robocall messages and secretary conversation is limited. Hence building such a system to perform with high accuracy with limited data is challenging. Moreover, having the system succeed in an adversarial environment makes it even more challenging. We like to think of the virtual assistants we propose as a hybrid conversational agent which uses rules

to make a casual conversation and at the same time is aimed with the goal of classifying a caller as human or robocaller.

2.6 Summary

Spam calls are a significant problem for telephone users. Unlike email spam, spam calls demand immediate attention. When a phone rings, a call recipient generally must decide whether to accept the call and listen to the call. After realizing that the call contains unwanted information and disconnecting the call, the recipient has already lost time, money (phone bill), and productivity. A study by Kimball et al. [77] found that 75% of people listened to over 19 seconds of a robocall message and the vast majority of people, 97%, listen to at least 6 seconds. Even when the recipient ignores or declines the call, today spammers can send a prerecorded audio message directly into the recipient's voicemail inbox. Deleting a junk voicemail wastes even more time, taking at least 6 steps to complete in a typical voicemail system. In recent work that conducted a large scale user study [78], it was shown that a significant fraction of users fall victim to telephone scams including robocalls. Therefore, an effective robocall defense is critical. We explore defenses for increasingly sophisticated threat models. We start by exploring the efficacy of blacklists and then investigate how automated voice interaction can be used to detect and stop robocalls without interrupting users who received unwanted calls.

CHAPTER 3

MEASURING THE EFFECTIVENESS OF PHONE BLACKLISTS

3.1 Introduction

In absence of public details on how current phone security apps work internally, in this chapter we explore different approaches for building a phone blacklist. Specifically, we build five different blacklists using the four datasets described in section 3.2, and then evaluate their effectiveness in blocking future unwanted or abusive calls in Chapter 4. Our blacklisting system architecture is shown in Figure 3.5.

In an effort to combat the increasing number of unwanted or fraudulent phone calls, a number of call blocking applications have appeared on smartphone app stores, some of which are used by hundreds of millions of users (e.g., Truecaller, Youmail). Additionally, a recent update to the default Android phone app alerts users of suspected spam calls [24]. However, little is known about the methods used by these apps to identify malicious numbers and how accurate and effective these methods are in practice. As we will show in Chapter 4, we empirically verified that our blacklists resemble third-party blacklists, and can therefore be used as a proxy to assess the effectiveness of proprietary phone blacklists.

We systematically investigate multiple data sources that may be leveraged to automatically learn phone blacklists, and explore the effectiveness of such blacklists by measuring their ability to block future unwanted phone calls. Specifically, we consider four different data sources: user-reported call complaints submitted to the Federal Trade Commission (FTC) [25], complaints collected via crowd-sourced efforts, such as 800notes.com and MrNumber.com, call detail records (CDR) from a telephony honeypot [26] and honeypot-based phone call audio recordings. We provide a detailed analysis of how such data sources could be used to automatically learn phone blacklists, measure the extent to which these

different data sources overlap, explore the utility of call context for identifying spam campaigns, and evaluate the effectiveness of these blacklists in terms of unwanted call blocking rates.

In performing this study, we are faced with a number of challenges, which we discuss in detail throughout this chapter. First, the data sources may contain noise. For instance, user-provided reports are often very short, written in a hurry (using abbreviations, bad grammar, etc.) and may contain incomplete or incorrect information, making it challenging to automatically infer the context of spam/scam calls. In addition, some data sources provide very limited information. For instance, due to privacy concerns, the user-reported FTC complaints are anonymized, and only report the time at which each complaint was submitted and the phone number the user complained about; the content or description of the complaints are not available to the public. Partial call context can be obtained from transcripts of recordings of calls made to honeypot numbers. However, recording calls faces legal hurdles, can be costly, and the quality of the recorded content may depend on the extent of caller engagement.

Because the utility of phone blacklists is based on blocking calls which are based on the calling phone number, another important challenge is represented by caller ID spoofing. As spoofing becomes more pervasive, the effectiveness of blacklists could be entirely compromised. Technical proposals have been published on caller ID authentication [53, 51, 52], and efforts have been put in place by telecommunications providers to make caller ID spoofing harder [29]. For instance, the industry-led "Strike Force" effort [28] has suggested numerous steps that can help mitigate spoofing (e.g., carriers can choose to not complete calls that originate from unassigned phone numbers). Also, removing the caller ID altogether can be detrimental for attackers, as users are less likely to answer calls coming from a "private" number. Because the cost of acquiring ever new, valid phone numbers is non-negligible, the effectiveness of phone blacklists could increase significantly, once these anti-spoofing mechanism are more widely deployed. Therefore, studying how phone

blacklists can be automatically learned, and evaluating their effectiveness, is important for both current and future telephony security applications.

In summary, we make the following contributions:

- We present the first systematic study on estimating the effectiveness of phone blacklists. We first analyze the characteristics of multiple data sources that may be leveraged to automatically learn phone blacklists, and then measure their ability to block future unwanted phone calls.
- We investigate a number of alternative approaches for building phone blacklists. In particular, we propose methods for learning a blacklist when call context (e.g., complaint description or phone call transcripts) is available, and when context is missing.
- We evaluate the effectiveness of the phone blacklists we were able to learn, and show that they are capable of blocking a significant fraction of future unwanted calls (e.g., more than 55% of unsolicited calls). Also, they have a very low false positive rate of only 0.01% for phone numbers of legitimate businesses.
- To link phone numbers that are part of long running spam campaigns, we apply a combination of unsupervised learning techniques on both user-reported complaints as well as from phone call audio recordings. We then identify the top campaigns from each data source, and show how effective blacklists could be as a defense against such campaigns.

3.2 Data Collection

3.2.1 Data Sources

Although commercial phone blacklisting services and apps do not openly reveal how their blacklists are constructed, some of the data sources they use to derive the blacklists are known or can be inferred. For instance, Youmail appears to leverage user complaints sub-

mitted to the FTC[1], whereas Baidu.com leverages online user complaints[2]. Other telephony security companies, such as Nomorobo [37] and Pindrop [79], leverage phone honeypot data [26].

To estimate the effectiveness of phone blacklists, we therefore use a multi-source data-driven approach that aims to gather and analyze datasets that are similar to the ones collected by commercial phone security services. Specifically, we consider two main sources of telephony abuse information: (i) phone call records collected at a large phone honeypot and (ii) unwanted call complaints submitted voluntarily by users. For each of these information sources, we assemble two different datasets (described below), which we divide into *context-rich* and *context-less* datasets. We say that a phone call record is *context-rich* if a recording or description of the *content* (i.e., the actual conversation that took place) of the phone call is available, along with metadata such as the caller ID, the time of the call, etc. Conversely, when only the metadata (i.e., no content) related to the phone call is available, we refer to the phone call record as *context-less*.

It is important to notice that, because we harvest phone call information from honeypots and user complaints, our datasets naturally contain only records linked to *abuse-related, unwanted, or otherwise unsolicited phone calls* (though a small amount of noise may be present, as discussed below).

3.2.2 Context-Less Phone Abuse Data

FTC dataset (FTC) - The Federal Trade Commission collects voluntarily provided user complaints about unwanted or abusive phone calls (e.g., robocalls, phone scams, etc.). Along with the reported call metadata, users can include a description of the related phone conversation. However, the data publicly released[3] by the FTC only contains the source

[1]The results of a reverse phone look-up via youmail.com include the number of FTC and FCC reports related to the queried number.

[2]For example, searching for Chinese spam-related phone numbers on baidu.com will return a brief report that includes the number of users who have complained about the queried number.

[3]Via a Freedom of Information Act request.

phone number and the timestamp of the reported call, and does not provide any information about the destination number (i.e., the user's phone number) or the content of the call itself, due to privacy reasons. From February to June 2016, we collected around 1.56 million complaints regarding 300,000 different source phone numbers.

Honeypot call detail records (CDR) - The CDR dataset contains detailed information about the calls coming into a telephony honeypot. It records the source phone number that made the call, the destination to which the call was made, and the time of the call. However, it does not provide the context of the call. This dataset contains records for more than one million calls received between February and June 2016 from approximately 200,000 distinct source numbers to approximately 58,000 distinct destination numbers, which are owned by the honeypot operator.

3.2.3 Context-Rich Phone Abuse Data

Crowd-sourced online complaints (COC). This dataset contains the online comments obtained from popular online forums, such as 800notes.com, MrNumber.com, etc., between Dec 1, 2015 and May 20, 2016. However, we only considered comments made between February to June so that this period overlaps with the time frame of the honeypot transcripts dataset (described below). The dataset contains about 600,000 actual "raw" complaints filed by users, containing the phone number that made the unwanted call, a timestamp, and a description of the content of the call. Since the comments are entered manually by frustrated users, the text describing the content of the call is typically quite noisy, as it contains many misspellings, grammatically inaccurate sentences, expletives, and in some cases apparently irrelevant information. It is also possible that the phone number and timestamp provided by the users could be mistyped.

Honeypot call transcripts (HCT) - This dataset contains about 19,090 audio recordings from 9,434 distinct phone numbers, extracted from a subset of calls made to the honeypot from February 17, 2016 to May 31, 2016. When a call is selected for recording, the re-

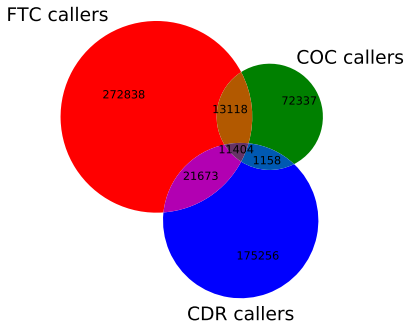


Figure 3.1: Overlap of callers across data sources.

sponder software attempts to emulate human behavior, to elicit a short conversation from the caller, which may allow for inferring the nature of the call. However, we noticed that in many cases the conversation attempt by the honeypot responder is irrelevant, in that the caller often simply plays a recorded message (i.e., a *robocall*). Calls to be recorded are selected using two different strategies: *random* and *targeted*. For random recordings, one in every ten calls were recorded. Targeted recordings were performed for a small amount of time, during which only calls coming from specific sources were recorded. The target source numbers were selected among the COC dataset complaints related to *tech support* and *IRS/tax* scam campaigns. The audio recordings were automatically transcribed using Kaldi [80]. Each dataset entry contains the time of the call, the source phone number, the destination phone number and the transcript of the call.

3.2.4 Data Volume and Overlaps

In the following, we present a number of high-level insights that can be gained from the four datasets described earlier. These insights help us understand how each dataset contributes to intelligence about telephony abuse, and what data source may first observe certain types of abuse.

Figure 3.1 depicts the overlap observed among callers across the FTC, CDR, COC datasets. Notice that, by definition, the source numbers in the HCT dataset are a small subset of the CDR dataset (see subsection 3.2.3). Interestingly, we found that many phone

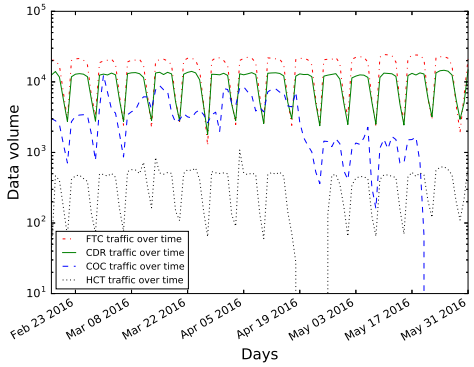


Figure 3.2: Data volume over time

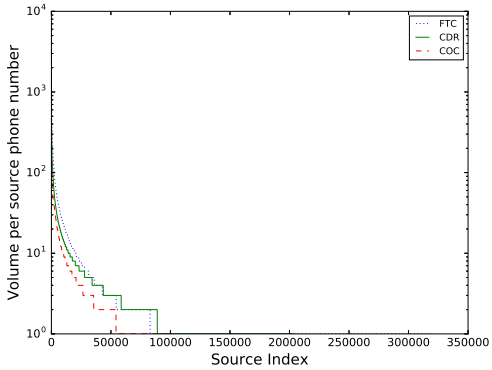


Figure 3.3: Data volume per source

numbers that call into the honeypot are never seen in the COC or FTC datasets. We suspect this may be due to the fact that many of the honeypot phone numbers were previously business-owned, which were returned and repurposed. Hence, the scammers/spammers targeting businesses tend to be captured more frequently, whereas spam targeting individual users is less commonly observed by the honeypot. This hypothesis is supported by a manual analysis of the transcripts obtained from the HCT dataset, which revealed the prevalence of business-oriented abuse. At the same time, since complaints collected by the FTC and COC datasets come from individuals, they tend to mostly reflect scammers/spammers targeting ordinary citizens (more details are provided in subsection 4.4.1).

Figure 3.2 reports the data volume (i.e., the number of calls or complaints) over time, across the four datasets. The periodic drops are due to lower call volumes during weekends. The drop in the HCT traffic between April and May is because call recordings were stopped due to operational issues during that particular period. Similarly, operational issues affected the collection of COC data towards the end of May.

Figure 3.3 shows that a large fraction of source numbers receive only one or few complaints, or perform only few honeypot calls. This may be due to a combination of factors, including caller ID spoofing and noise due to misdialing, with spoofing being the most likely and prevalent culprit.

Figure 3.4 shows the difference in days between when the honeypot received the first

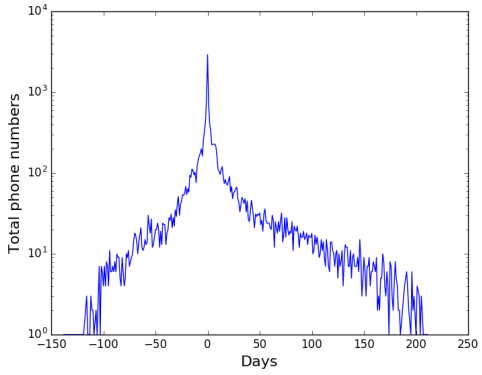


Figure 3.4: Timeliness – CDR vs. COC

call from a given source phone number and the time when the first complaint about that same phone number appeared in the COC dataset. Among the phone numbers that are observed in both the CDR and COC datasets, 20% of them were seen on the same day, whereas 35% of of the numbers were seen in the honeypot before they were complained about by users.

3.2.5 Gathering Ground Truth

Ideally, the datasets we collected should be free of noise. Indeed, both the honeypot records and the voluntary user complaints are by nature related to abusive or unwanted calls. However, as mentioned earlier, the datasets may contain noise, for instance due to misdialed calls to the honeypot or mistyped phone numbers in the complaint reports.

Establishing the true nature of source phone numbers that appear in our datasets is challenging, as no single authoritative entity exists that can certify whether a certain phone number is being used for legitimate or malicious activities. We have therefore reverted to taking a conservative, best effort approach for ground truth collection based on multiple third party providers. Specifically, we leverage reverse phone lookup services provided by Whitepages [81], YouMail, and TrueCaller, to obtain independent insights about the nature of a fraction of the phone numbers we observed.

Query results from the above third parties contain information on whether a number is

believed to be a spam/scam-related number. While we have no detailed information about how these third-party services classify phone numbers, public documentation suggests that they leverage user complaints, along with other data points. As these are third-party commercial systems with a large user base (millions of mobile application downloads), we believe it is reasonable to assume that they have checks in place to limit false positives to a minimum, because high false positives may otherwise deter app/service adoption and revenue. Therefore, if a source number is reported by these services as *spam*, we consider the label to be correct (unless disputed via other means).

Whitepages additionally provides information such as whether a phone number is likely a VOIP, toll free, mobile or landline number; it also indicates whether the number is used for commercial purposes, and provides owner information such as name, street address, etc., when available.

In addition to information on phone numbers likely involved in spam activities, we also collect a large set of phone numbers that can be considered as legitimate (i.e., non-spam), by crawling the YellowPages.com phone book. We later leverage this set of phone numbers to estimate the false positive rate of our phone blacklists, as explained in section 4.7. For instance, we may consider phone numbers that appear to have valid owner information as likely benign numbers, whereas phone numbers whose owner information appear to have been forged (e.g., a clearly fake name and/or address) could be confirmed as likely abusive (remember that by nature our data sources already collect numbers that are highly likely abusive).

In addition to collecting ground truth from third-parties, in some cases we attempted to verify the nature of phone numbers that are candidates for blacklisting by calling them back. For instance, for blacklisted numbers for which context is available (e.g., for numbers related to call transcripts), calling back allows us to verify whether the content of our call is similar to the previously recorded context.

3.2.6 Ethics

The telephony honeypot is operated by a commercial entity, and raw CDR data was accessed under non-disclosure agreements. The honeypot is programmed to record only (a small subset of) phone calls that meet rigorous legal requirements, according to US federal and state laws.

The FTC data was obtained in response to a freedom of information act (FOIA) request, and does not contain any sensitive information. For instance, the FTC does not disclose the destination phone numbers and user complaint descriptions, to protect the privacy of the reporting users. The FTC currently makes this data available freely to all parties.

The ground truth data collected from third-party sources, such as YouMail, TrueCaller, and Whitepages, is limited to publicly accessible information. To increase the number of available queries, we used the Whitepages Premium service. For all Whitepages reverse phone lookups, we carefully refrained from collecting sensitive information from background reports (i.e., we never analyzed or stored any information about bankruptcies, liens, arrest records, family members, etc., which is available from Whitepages Premium).

When calling back numbers that are candidate for blacklisting, we only called those that asked to be called back, according to the honeypot transcripts in the HCT dataset. Furthermore, when calling back we never interacted with a real human. Every call we made went through an automated interactive voice response (IVR) system.

We did not involve human subjects in our research. The honeypot calls were recorded by a third-party company, while abiding by US laws (e.g., single-party consent requirement). Calls made by us were limited to interactions with automated IVR systems. Because of these reasons, we did not seek explicit IRB approval.

3.3 Phone Blacklisting

Blacklisting has been extensively studied as a way to defend against Internet abuse [17, 82, 20, 83]. For instance, domain name, URL, and IP address blacklists are commonly used to defend against email spam, phishing, and malware infections [84, 85, 17]. Recently, phone blacklisting has started to make its way into real-world applications [23, 22]. However, to the best of our knowledge, the effectiveness of blacklisting approaches to defend against abuse in the telephony domain has not yet been systematically studied.

3.3.1 Example Use Case

We consider a scenario in which smartphone users[4] install an app that implements the following functionalities: the app is notified every time a phone call is received, it checks the caller ID against a phone blacklisting service[5], and informs the user on whether the calling phone number is believed to be used for phone spam/abuse activities. This use case is similar to currently available popular apps [22, 22, 87].

Depending on user preferences, the app may strictly enforce the blacklist, and immediately block the call [86] (while still notifying the user of the event, for example). Alternatively, the user may opt for a *soft blacklisting* enforcement, whereby the user is provided with information about if/why the calling number was included in the blacklist and will have to decide whether to pick up the call or not [87]. For instance, the user may be informed that the calling number was previously complained about by other users (e.g., via the FTC complaints service). If context is available (see subsection 3.2.3), the app may also provide information about a specific (set of) spam campaign in which the number has been involved.

[4] Blacklisting services may also be used by telephone networks to defend landline phones, for instance. While in the past there existed strict regulatory constraints that may have prevented carriers from using blacklists to block calls, such restrictions seem to have been recently relaxed [86].

[5] We also assume that queries to the blacklisting services can be done securely and in a privacy-preserving way, similarly to URL blacklists such as Google Safebrowsing.

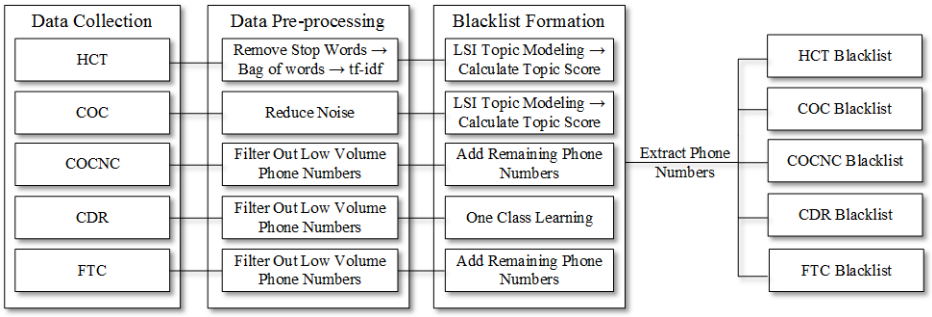


Figure 3.5: System Architecture

3.3.2 Context-less blacklisting

As discussed in section 3.2, the FTC and CDR datasets do not include the context of a call. To build a blacklist based on these context-less data, we therefore focus on identifying anomalies in calling patterns.

Before we describe our blacklisting approach, it is important to remember that, by nature, the FTC and CDR datasets contain only information about unwanted or abusive calls. While we cannot exclude the presence of small amounts of noise (e.g., due to misdialed calls captured by the honeypot, or numbers incorrectly reported to the FTC), it is reasonable to assume the fraction of noisy reports/calls is small. We leverage this as the main observation to guide our approach to phone blacklisting in absence of context.

Blacklisting using the CDR data

Because the CDR dataset may (by chance) collect misdialed phone calls, we first apply a pre-filter step by removing phone numbers that, during the training data collection period, made less than θ_c calls to less than θ_d destination honeypot numbers. In other words, we only consider a phone number for the next processing steps if it made more than θ_c calls and contacted more than θ_d different destinations within a predetermined observation time (in our experiments, we primarily use $\theta_c = 5$ and $\theta_d = 3$, but also perform additional experiments that show how the blacklist effectiveness varies with these parameters). Notice that this pre-filtering step is fairly conservative, and that source phone numbers actively

involved in spam activities will tend to pass this simple filter.

To build the CDR-based blacklist, we analyze the behavior of the remaining source phone numbers. For each of the source phone numbers, p_i, we compute a *blacklist score*:

$$s(p_i, \Delta t) = \alpha \times vol(p_i, \Delta t) + \beta \times nod(p_i, \Delta t) \tag{3.1}$$

where $vol(p_i, \Delta t)$ is the number of calls made by p_i within time Δt, whereas $nod(p_i, \Delta t)$ is the number of destination numbers called by p_i in the same time period, and α and β are tunable parameters.

As spammers typically tend to reach a large number of potential victims, we set the value of β greater than α (in our experiments, we set β=0.2 and α=0.1). Any number p_i whose blacklist score $s(p_i, \Delta t)$ is greater than a threshold θ_b, which is learned from past observations, is added to the blacklist.

To learn the blacklist, we use a *one-class* learning approach [88, 89]. This choice of learning paradigm is guided by the challenges in collecting ground truth labels (see subsection 3.2.5), especially for benign phone numbers. To identify spam-related phone numbers within the CDR dataset, which we then leverage for training the blacklisting threshold, we proceeded as follows. Given the set $\mathcal{P}_{CDR}$ of all source phone numbers calling into the honeypot (excluding the pre-filtered numbers), we find the intersection between these numbers and the phone numbers reported in the FTC and COC datasets during the observation period Δt. Because these datasets are collected in a completely independent way (honeypot calls vs. user complaints), we assume that phone numbers that appear in two or more datasets are the most likely to actually be spam-related. For instance, if a number p_j called into the honeypot multiple times (enough to pass the pre-filter), and multiple users, in a completely independent way, complained about the same p_j number via the FTC portal, we label p_j as *spam*. We then use this *one-class* labeled subset of spam numbers, $P_s \in (\mathcal{P}_{CDR} \cap \mathcal{P}_{FTC})$, to learn the θ_b threshold. Specifically, we sort the number in P_s

by their blacklist score (see Equation 3.1), and set θ_b so that the top 99% of all numbers, by score value, are added to the blacklist. In the spirit of one-class learning [89, 88], the remaining 1% of numbers are considered to be *tolerable* false negatives, and have the benefit of making the decision threshold sufficiently "tight" around the bulk of spam-labeled data to filter out the possible remaining dataset noise (i.e., potentially benign numbers that accidentally called into the honeypot).

Blacklisting using the FTC dataset

The FTC dataset is the largest in terms of volume of reported phone numbers, compared to the other datasets. As mentioned in section 3.2, the information provided by the FTC is very limited, as it only contains the user-reported phone numbers and a timestamp for each complaint report. Unlike the CDR dataset, no information is provided regarding the destination numbers.

Like the CDR dataset, the FTC dataset may also contain small amounts of noise, for instance due to a calling number being typed erroneously into a user complaint. To filter out this possible noise, we exclude all phone numbers that have been reported in less than θ_c complaints (notice that this parameter is similar to the θ_c filtering threshold used for the CDR-based blacklisting). All remaining numbers are then simply added to the blacklist. The reason why we do not perform any additional filtering is that the FTC dataset contains official complaints that users send to the FTC; as such, this dataset intuitively tends to contain lower amounts of noise, compared to the CDR dataset.

Context-less blacklisting using the COC dataset

For comparison purposes, we apply the same process described above for the FTC dataset to the COC dataset, pretending that no context is available. In other words, from the user complaints in the COC dataset we only extract the timestamp and the reported phone number (i.e., the source numbers users complained about), and apply the filtering approach

described above for the FTC complaints. In the remainder of this chapter, we refer to this blacklist as the COCNC blacklist, where NC stands for *no-context*.

3.3.3 Context-rich blacklisting

The context of a call, when available, can be used to understand the nature and content of the conversation, and provide more definitive indication on whether a call is potentially an unwanted or fraudulent one. For example, calls with similar context can be clustered together to discover spam campaigns, and the phone numbers related to the campaigns can be added to the blacklist, along with contextual information. Therefore, when a user receives a call from a number that belongs to a context-rich blacklist, we could not only inform the user that the incoming call is likely unwanted or abusive, but also provide a short description (e.g., via the example app described in subsection 3.3.1) of what kind of spam campaigns the number has been involved within the past. This information may be particularly useful when a *soft blacklisting* approach is selected, as it may help the user make a decision on whether to pick up the call or not.

Blacklisting using the HCT dataset

To derive a phone blacklist based on the honeypot call transcripts (HCT) dataset, we take the following high-level steps: (i) we perform transcript text analysis using topic modeling via latent semantic indexing (LSI) [90], to extract possible campaign topics; (ii) we then label transcripts based on their most similar topic, and group together calls that likely belong to a common spam campaign; (iii) finally, phone numbers belonging to a spam campaign are added to the blacklist. Below, we provide more details on this blacklisting process.

We first use a data pre-processing phase, which aims to filter out possible noise from the transcripts (e.g., noise due to imprecise speech transcription). To this end, we use the following steps: (1) *stop-words* are removed and a dictionary of the remaining terms is extracted from the transcripts' text; (2) each transcript is then converted into a bag of

words, and each word is assigned a score using TF-IDF [91]. These two steps transform each call transcript into a vector of numerical features (i.e., a feature vector).

Table 3.1: LSI topic modeling on HCT – top 10 topics

topic 0	google, listing, front, page, business, verify, press, removed, searching, locally
topic 1	cruise, survey, bahamas, awarded, correctly, included, participate, short, congratulation, selected
topic 2	listing, verify, front, google, page, updated, record, show, end, list
topic 3	verification, address, name, phone, number, cancel, flagged, map, notice, business
topic 4	hotel, pressed, exclusive, telephone, husband, marriott, detail, announcement, pre, star
topic 5	hotel, exclusive, husband, marriott, star, stay, placed, complimentary, further, telephone
topic 6	electricity, bill, per, system, stop, increase, energy, renewable, soon, coming
topic 7	optimize, found, date, order, indicate, critical, online, updated, show, end
topic 8	system, interest, eligibility, cost, account, rate, credit, notice, card, lower
topic 9	business, interest, eligibility, thousand, application, loan, rate, bad, system, qualifies

We then use a topic modeling approach on the feature vectors obtained from the steps mentioned above. Let Δt be a data observation window, and $\mathcal{H}(\Delta t)$ be the set of call transcript feature vectors obtained during Δ_t. We use LSI, a natural language processing technique that leverages SVD [92] to map documents (i.e., transcripts, in our case) from a syntactic space (the bag of words) to a lower-dimensional semantic space represented by a (tunable) number τ_{hct} of *topics*. In concrete terms, each topic is represented by a set of representative keywords that may be interpreted as describing a campaign theme. Table 3.1 shows the top 10 topics (sorted by eigenvalue) extracted from our HCT dataset (more details on the experimental results are provided in the next chapter).

At this point, each transcript can be represented as a weighted[6] mix of topics, rather than a set of words [90]. Among these, we can identify the topics with the highest weight, which can be interpreted as indicating what spam campaigns the calling number recorded in the transcript is involved with.

The LSI algorithm requires as a parameter the desired number of topics to be kept in

[6]We consider absolute values for the topic weights.

the SVD decomposition. Choosing the best value for the number of topics is often done either manually, by leveraging domain knowledge, or by measuring topic coherence [93]. However, coherence measures are themselves still a subject of research in the machine learning domain, and don't always bring to satisfactory results in practice. Therefore, in this chapter we revert to manual selection driven by empirical results, and leave a fully automated selection of the optimal number of topics to future work. In our experiments, we first set the maximum number of LSI topics to 50. Then, once the topics are extracted, we manually analyze them and mark the ones whose keywords more clearly indicate a spam campaign, whereas the other topics are effectively discarded from a transcript's topic mixture vector. As a byproduct, this manual analysis also has the advantage that it allowed us to associate a human-interpretable campaign *theme* to each of the remaining topics. For instance, we summarize `topic 0` in Table 3.1 as the *Google Listings* spam campaign (notice that when analyzing a topic, not only we can refer to the topic's keywords, but also to the full text of the transcripts that are associated with that topic with a high weight).

At this point, we have a set of topics, $\mathcal{T}$, that are labeled with a relevant campaign theme, and we aim to do two things: (1) decide what source numbers for the transcribed calls should be blacklisted; and (2) leverage the topic model to group together call transcripts, and related source phone numbers, that likely belong to the same spam campaign. To this end, we first compute a *topic similarly* score $S_{i,j} = S(tr_i, \tau_j)$ that indicates how strongly a transcript tr_i is associated to each topic $\tau_j \in \mathcal{T}$. We calculate the topic scores by normalizing the topic weights output by the LSI topic modeling algorithm. Specifically, for every transcript tr_i and every topic τ_j, the topic modeling algorithm will assign a weight $w_{i,j} = w(tr_i, \tau_j)$. We compute the normalized weights for each transcript as $w'_{i,j} = |w_{i,j}| / \sum_j |w_{i,j}|$, and set the score $S_{i,j} = w'_{i,j} \in [0, .., 1]$.

To decide whether a source phone number p_i responsible for call transcript tr_i should be blacklisted, we proceed as follows. We first compute the topic most similar to tr_i, namely $k = \arg\max_j(S_{i,j})$. Then, if $S_{i,k}$ is greater than a predetermined threshold θ_k, we assign

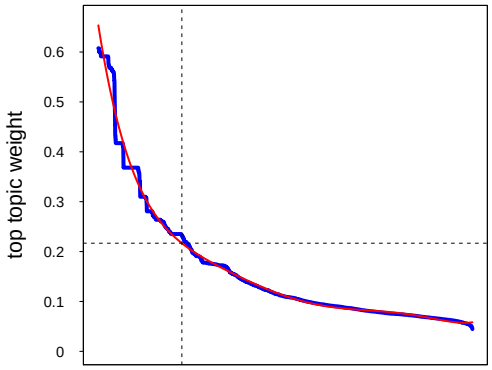


Figure 3.6: HCT blacklist score threshold learning

p_i to the HCT blacklist. The threshold θ_k is learned as follows. Let S_i^* be the highest topic score for transcript tr_i. We first plot the distribution of scores S_i^* computed over all transcripts, as shown in Figure 3.6 and then compute θ_k by finding the "knee" of the curve (the knee finding process is explained in details later in this chapter).

Now, for every blacklisted number p_i, we have the topics that are most similar to the transcripts related to p_i, and can therefore label p_i with one or more campaigns themes (an analysis of campaigns themes and related blacklisted numbers is reported in section 4.8).

Blacklisting using the COC dataset

Like honeypot call transcripts, user comments from online forums such as 800notes.com, MrNumber.com, etc., also provide us with the context of an unwanted call. However, transcripts and user comments datasets are different in nature, as user comments only provide a user's version – a subjective textual description – of the content of a call. To derive a blacklist using the COC dataset, we follow a process very similar to the one we used for the HCT data, with some small changes that take into account differences in the nature of the two data sources.

Via manual analysis of a few initial samples of online complaints data, we noticed that

user-provided descriptions of unwanted calls tend to be noisier in nature than transcripts from call recordings. This is fairly intuitive: while the transcripts faithfully reflect the conversation (often represented by a well-played recorded spam message), user complaints typically consist of high-level descriptions of a call, in which abbreviations, bad grammar, and expletives are used to express discontent. Therefore, to reduce noise we use a more stringent pre-processing step, compared to the HCT dataset. First, we only consider phone numbers that were complained about at least θ_{coc} times ($\theta_{coc} = 10$, in our experiments). We also remove stop words and punctuation from the comments, and combine all comments about a single phone number into a single text document. This latter aggregation step is motivated by the following considerations: (1) complaints from different users that receive calls from the same phone number are often similar, because the number is used to perpetrate the same spam campaign by calling multiple destinations; (2) online complaints are often very short, making it difficult to automatically analyze them independently from each other; (3) by aggregating multiple complaints, we obtain larger documents that can be more easily analyzed using topic modeling, with a process similar to the one described in subsubsection 3.3.3.

Let $\mathcal{C}(\Delta t) = \{c_{s_1}, \ldots, c_{s_n}\}$ be the set of complaint documents, where document c_{s_i} is an aggregate (i.e., a concatenation) of all user complaints about calling number s_i observed during period Δt. As for the HCT blacklist, we apply LSI on $\mathcal{C}(\Delta t)$ to derive a set of possible spam campaign themes, and at the same time associate each calling number (via the related complaints) to a mix of topics, as done for the HCT blacklist. We then decide what source numbers for the transcribed calls should be blacklisted by computing the topic scores, plotting the distribution of the maximum score, and computing the blacklisting score threshold by finding the "knee" of this distribution.

Finding curve knee

To find the "knee" of a topics weight curve , we use the following algorithm, which is visually represented in Figure 3.7:

1. Plot graph of sorted highest topic weights (blue curve);

2. Fit a low-order (e.g., order 4 or 6) polynomial onto the curve (red curve);

3. Compute the intersection between the left and right tangent lines (gray lines with negative slope);

4. Find the line that bisects the angle between the two tangents (gray line with positive slope);

5. Find the point in which the bisection line intersects the polynomial;

6. Project this point onto the y axis (dashed horizontal line).

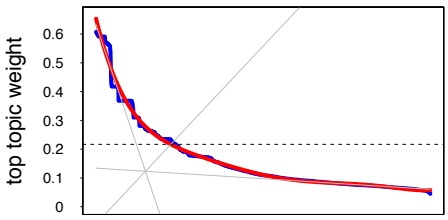


Figure 3.7: Finding curve knee

Even though both context-rich and context-less blacklists consist of phone numbers, each phone number in the context-rich blacklist is associated with a label derived from its context. The label identifies the type of scam campaign for which the phone number was used. This label could be used to inform a user about the potential content of a suspected spam call.

3.4 Conclusion

Call blocking apps for smartphones are now becoming commonplace but little is known about the efficacy of such applications in protecting users from unwanted/scam calls. We present results of a data-driven study that utilizes multiple data sources to explore the feasibility and effectiveness of phone blacklists for blocking such calls. We demonstrate how

phone blacklists can be learned both when the context of a call is known and when such context is not available due to a variety of reasons (e.g., privacy concerns, recording overhead etc.). In the next chapter, we evaluate the effectiveness of phone blacklists constructed with the techniques discussed in this chapter.

CHAPTER 4
EVALUATING PHONE BLACKLISTS

4.1 Introduction

In this section, we evaluate the effectiveness of the phone blacklists we constructed following the methods described in the previous chapter. We first show that our blacklists are representative of real-world phone blacklisting applications, by vetting them against two commercial third-party telephony security services. In addition, we perform an analysis of how blacklists formed from different data sources can complement each other. We then assess how well the blacklists we construct can help to block future spam (i.e., unwanted or abusive) phone calls, by evaluating each blacklist in terms of their *call blocking rate* (defined below). Finally, we analyze a few prominent phone spam campaigns, and demonstrate how effective the blacklists would be in blocking these campaigns.

4.2 Call blocking rate (CBR) definition

We evaluate the effectiveness of a phone blacklist based on its ability to block future unwanted call. Therefore, to enable the evaluation we first need to more formally define the concept of blocking rate. Given a blacklist $\mathbb{B}(\mathcal{D}, \Delta t) = \{p_1, \ldots, p_m\}$ containing m phone numbers learned from dataset $\mathcal{D}$ over a *training* observation period Δt, we consider the set $C(\lambda_t)$ of calls (or complaints, depending on the blacklist being analyzed) observed at a future *deployment* time period λ_t. We then compute the ratio $r(\mathbb{B}, \lambda_t) = N_{bl}(\lambda_t)/N_{tot}(\lambda_t)$ between the number of calls (or complaints) $N_{bl}(\lambda_t)$ that would have been blocked by the blacklist $\mathbb{B}$, and the total number of calls (or complaints) $N(\lambda_t) = |C(\lambda_t)|$ received during period λ_t.

In the remainder of this section, the set $C(\lambda_t)$ will represent either the set of calls

received by the honeypot, as recorded in the CDR dataset, or user complaints from the FTC or COC datasets, depending on the blacklist that is being evaluated. In the first case, we refer to $r(\mathbb{B}, \lambda_t)$ as the *call blocking rate*, whereas in the second case we refer to it as the *complaint blocking rate* – both abbreviated as CBR.

In the case of the CDR data, we essentially pretend that the phone numbers belonging to the honeypot are owned by users, and consider a future honeypot call to be *blocked* if the related calling phone number was included in $\mathbb{B}(\Delta t)$. Therefore, the CBR estimates the fraction of future unwanted calls towards real users that would be prevented by the blacklist. In addition, in this case we can also measure how many users would be defended against spam calls, by counting the number of distinct destination numbers that thanks to the blacklist did not receive the unwanted calls.

In the case of the blacklists derived from the FTC and COC datasets, the CBR measures the fraction of future complaints that would be prevented by the blacklist. Computing the number of *blocked complaints* is motivated by this simple observation: if an app enforcing the blacklist was widely deployed, or telephone carriers directly blocked calls based on the blacklist, users would not receive any more unwanted calls from the blacklisted numbers, and would therefore stop complaining about them. Thus, the number of complaints that would be prevented (i.e., blocked) is a reflection of the effectiveness of the blacklist.

4.3 Experimental Setup

Our experiments for computing the CBR are performed as follow, for all the datasets and blacklisting approaches described in section 3.2 and section 3.3. Let $\mathcal{D}$ be a dataset of calls or complaint records (e.g., the CDR or FTC dataset). We start by setting an initial training period Δt_0 of one month, corresponding to the first month of data collected in $\mathcal{D}$. We use this first month of data to learn the first blacklist $\mathbb{B}_0 = \mathbb{B}(\mathcal{D}, \Delta t_0)$. We then consider one day, λ_{t_0}, of data from $\mathcal{D}$ collected on the day immediately after period Δt_0, and compute the CBR $r_0 = r(\mathbb{B}(\mathcal{D}, \Delta t_0), \lambda_{t_0})$.

We then set $\Delta t_1 = \Delta t_0 + \lambda_{t_0}$, thus extending the training period by one day, and compute $\mathbb{B}_1 = \mathbb{B}(\mathcal{D}, \Delta t_1)$ and $r_1 = r(\mathbb{B}(\mathcal{D}, \Delta t_1), \lambda_{t_1})$, where λ_{t_1} again represents the day after Δt_1. We repeat this process for all subsequent days of data available in $\mathcal{D}$, thus obtaining a series of blacklists $\{\mathbb{B}_i\}_{i=0..k}$ and related blocking rate estimates $\{r_i\}_{i=0..k}$. In other words, every day we extend our blacklist training set by adding one day of data from $\mathcal{D}$ to the previous training dataset, and then test the obtained blacklist against the following day of data in $\mathcal{D}$. This allows us to estimate how effective the blacklist would be in blocking future calls (or complaints) related to spam phone numbers we learned up to the previous day.

4.4 Characterizing Blacklisted Numbers

We now analyze the overlap among blacklists learned over different data sources, and discuss how our blacklists align with phone blacklists provided by third-party apps.

4.4.1 Overlap among our blacklists

Figure 4.1 shows the size (i.e., the number of blacklisted phone numbers) and overlap among the blacklists learned as discussed in section 3.3. These results are obtained by building the blacklists over the entire set of data available from each of our data sources. In other words, given a dataset $\mathcal{D}$, we consider the entire period of time Δt_{max} in which data was collected, and compute the related blacklist $\mathbb{B}(\mathcal{D}, \Delta t_{max})$.

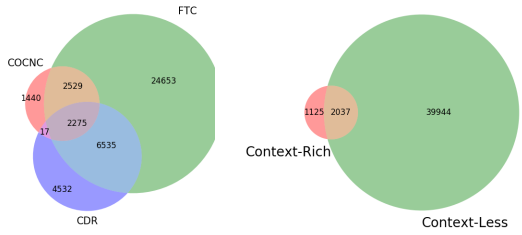


Figure 4.1: Blacklists size and overlap

As we can see from the figure, the overlap among blacklists derived from different data sources is limited. Specifically, there exists only partial overlap between spam phone numbers observed in the three main data sources, namely the FTC complaints, online user complaints (i.e., the COCNC blacklist) and honeypot call records (i.e., the CDR blacklist). This shows that the different blacklists provide coverage for different sets of phone numbers, and are therefore complementary to each other.

The differences between honeypot calls and user complaints is likely due to the particular nature of the honeypot-owned phone numbers. While user complaints mostly reflect spam campaigns that target generic users, many of the honeypot-owned phone numbers were previously owned by businesses, and tend to attract business-oriented phone abuse. We verified this by analyzing the honeypot transcripts in the HCT dataset, many of which are related to *Google business listing*, *business loans*, and other business oriented phone spam campaigns. On the other hand, user complaints tend to reflect more "traditional" spam campaigns, including *IRS* scam, *tech support* scam, *payday loans* scams, etc. (see section 4.8 for more details about spam campaign analysis).

We conducted further analysis to better understand the limited overlap between the COCNC and CDR blacklists. There are 2,292 overlapping phone numbers in these two blacklists, and we found that these numbers are not among the most complained about or heavy honeypot callers. This seems to refute the intuition that phone numbers that make the most honeypot calls are also more likely to be complained about by users. Again, this may be due to the different, business-oriented nature of the honeypot-owned numbers, as discussed above.

The FTC is the largest dataset in our analysis, and the blacklist constructed from the FTC complaints is the largest one, in terms of number of blacklisted phone numbers. Comparing the FTC blacklist to all other four blacklists combined shows an overlap of less than 50%. On the other hand, the context-rich blacklists are significantly smaller than the context-less ones. The main reason is that in the context-rich case a phone number is

added to the blacklist only if it can be associated to an identifiable spam campaign (see subsection 3.3.3).

4.4.2 Overlap with third-party blacklists

As discussed in subsection 3.2.5, we leverage third-party phone security and information services to gather independent (partial) ground truth on spam-related activities and characterize the phone numbers in our blacklists, as described below. We first assess the overlap between our blacklists and phone abuse information available from Youmail and Truecaller, and then use the Whitepages reverse phone lookup service to gather further insights on a subset of the numbers.

To estimate the overlap between our blacklists and third-party blacklists, we selected a random sample of 12,500 source phone numbers from all of our datasets, and performed reverse phone lookup queries. We found that 2.4% of the queried numbers were labeled as *spam* by Youmail. To determine the overlap between our blacklists and Youmail's, we proceeded as follows. If Youmail labeled a queried phone number as *spam*, we checked if the number was also present in our blacklists or not, and found that 87% of the phone numbers blacklisted by Youmail were also present in one or more of our blacklists. Most of the numbers labeled as *spam* by Youmail that were not included in our blacklists are present in our FTC dataset, but they were not included in our blacklist because they had a very low number of complains. This is in accordance with our attempt to be conservative, and filter-out possible noise in the user complaints, as explained in subsubsection 3.3.2. On the other hand, it appears that Youmail labels a number as *spam* even if only one user complained to the FTC about that number. If we had added all FTC-complained callers to our blacklist, we would have a 98% match of our blacklisted numbers against Youmail's blacklist. We also found that among the numbers that were not labeled as spam by Youmail, about 1% of them were present in our blacklists. These results show that, combined, our blacklists are representative of a commercial blacklisting app such as Youmail.

To compare our blacklists with Truecaller, we took a similar approach. In this case, we found that 38% of the numbers we queried were labeled as *spam* by Truecaller, and that only 13% of all the numbers labeled as *spam* by Truecaller were contained in our blacklists. The reason is that Truecaller seems to be labeling a number as abusive even if only one Truecaller user reported it as such. In fact, by labeling as *spam* only numbers that have been reported as abusive to Truecaller by at least 5 users, we found that 75% of these numbers are present in our blacklists. As in the previous analysis of Youmail, we found that of the numbers that were not labeled as spam by Truecaller, only 13% were present in our blacklists. The majority of this 13% of numbers matches our FTC blacklist, and are therefore reported in multiple user complaints.

The above results confirm that our blacklisting approach aligns fairly well with real-world, commercial phone blacklists (especially with the Youmail app), and can therefore be leveraged as a proxy to estimate how effective third-party phone blacklists are in defending real users from unwanted or abusive calls.

4.4.3 Analysis of phone number types

To further characterize the phone numbers included in our blacklists, we turned to the Whitepages [81] reverse phone lookup service. Whitepages is a third-party provider that gathers comprehensive information about phone numbers, including detailed phone ownership information, and whether the phone number *type* falls within one of the following categories: *VoIP*, *toll-free*, *landline*, or *mobile*.

As Whitepages' public querying interface only allows for a limited number of queries, we first started by selecting a sample of 150 phone numbers in the overlapping region across HCT, COC and CDR blacklists, and analyzed their query results. Because these numbers appeared in three different blacklists, they are among the highest confidence spam numbers in our datasets. We found that 67% of these numbers are VoIP numbers for which no owner information was available. This is not surprising, as it is common for abusive

calls to originate from VoIP numbers [94, 95]. The lack of owner information also suggests that these phone numbers are unlikely to belong to legitimate users. In addition, 5% of these numbers did not appear to have been assigned by any carrier. Surprisingly, only 22% of the numbers we queried were flagged as scam/spam callers by Whitepages itself. This suggests that Whitepages may be missing a large fraction of numbers that can be labeled as *spam* with high confidence.

We then expanded our sample of phone numbers by randomly drawing a total of 400 numbers from all blacklists and performing an analysis of the reverse lookup results. Out of all the phone numbers we queried to Whitepages, 71% of them were present in our FTC blacklist. Table 4.1 summarizes some of the results we obtained, where the *cdr*, *coc*, and *hct* represent results related to phone numbers that were included only in the CDR, COC, or HCT blacklist, respectively. Columns *hct/coc*, *hct/cdr*, and *coc/cdr* represent a random sample of the phone numbers that belong to the intersection between pairs of blacklists. Table 4.1 also report the percentage of phone number for which ownership information was present. As we can see, only a relatively small fraction of the (potentially spoofed) numbers appear to be owned by a valid user or business. In addition, we found that some owner information (e.g., owner name and location) is highly likely forged.

We can see that a phone number that is included in at least two of the blacklists is most likely to be a VOIP number. Moreover, only a few of the phone numbers included in at least two of the blacklists have any owner information, and most of the phone numbers for which we found owner information have incorrect information, for example, having gibberish or generic words like ”local”, ”support”, ”market” in the owner’s name field. The phone numbers that are found in only one blacklist also have a high probability of being a VOIP number and owner information is less likely to be found.

We have done further analysis on the phone numbers for which owner information was found. We went back to the datasets and checked if context is available for these callers and found evidence of them being involved in scam/spam campaigns like IRS, Tech support,

Table 4.1: Whitepages reverse phone lookup results

	cdr	coc	hct	hct/coc	hct/cdr	coc/cdr
VoIP	69%	37%	57%	80%	76%	70%
toll-free	9%	2%	0%	0%	2%	0%
landline	20%	16%	26%	20%	22%	30%
mobile	2%	45%	17%	0%	0%	0%
owner info	7%	10%	12%	2.5%	2%	5%

credit card services etc. We also checked the landline and tollfree numbers for context in the transcripts and online comments dataset, and whenever context was available, we found a significant number of complaints in the online comments or transcripts proving that the numbers are indeed involved in malicious activities. Our analysis thus provides strong evidence that blacklisted phone numbers are indeed involved in unwanted or abuse calls.

4.5 Evaluating Context-Less Blacklists

We now evaluate the call (or complaint) blocking rates, which we defined in section 4.2, for the CDR, FTC, and COCNC blacklists (see subsection 3.3.2).

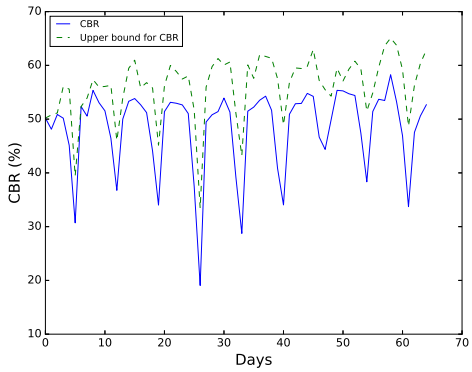


Figure 4.2: CDR call blocking rate

Figure 4.2 shows the call blocking rate when applying the CDR blacklist to future calls into the honeypot. The y axis depicts the percentage of calls to the honeypot on a given day that would have been blocked by using the most recent blacklist (i.e., the blacklist trained on data collected up to the previous day, as detailed in section 4.3). The blue curve show the results for our CDR blacklist (see subsubsection 3.3.2), whereas the green line shows

the upper bound for the effectiveness of the blacklist. This upper bound was obtained by adding *all* source phone numbers previously observed in the honeypot (during the training period Δt) to the blacklist, without applying any noise filtering or threshold learning, and then testing against the calls received on the next day (i.e., λ_t). The periodic call blocking rate drops in this graph are due to periodic appearance of new callers (every 7 days or so).

Although the blocking rate decreases during some days, by updating (i.e., re-training) the blacklist daily, over 55% of honeypot calls could be blocked using our CDR blacklist, on average. This shows that the blacklist can be fairly effective in blocking unwanted calls. At the same time, the upper bound curve clearly demonstrates that every day 40% or more of all calling numbers observed from the honeypot are always new (never seen before). Although we need to consider that noise (e.g., misdialings) may be recoded in the dataset and that spammers also churn through new valid numbers, it is likely that the large volume of new numbers is for the most part due to spoofing.

Figure 4.3 reports the complaints blocking rate (defined in section 4.2) for the FTC blacklist. These results were computed for three different values of the θ_c blacklisting threshold defined in subsubsection 3.3.2. Naturally, the lower the threshold, the higher the blocking rate, because more numbers will be added to the blacklist. In Figure 4.4, we report analogous results for the COCNC blacklist (subsubsection 3.3.2), including results for $\theta_c = 1$, which provide an upper bound for the effectiveness of the blacklist.

As we can see, from Figure 4.3, the CBR for the FTC blacklist is much higher than in the CDR and COCNC cases. On possible explanation for this is that that users may be reluctant to report an unwanted call to the FTC unless they receive multiple calls from the same number, given the official nature of the complaint to a government agency. In this case, they complaints would likely include more "stable" source phone numbers, and naturally filter most of the spoofed calls. Another factor to consider is that not all users will complain to the FTC; namely, for a number to appear into the FTC dataset, it is likely that several users received a call from the same number, but only one or few users decided to

formally complain.

Unlike the FTC dataset, the CDR dataset includes all (unanswered) calls to the honeypot, even if a number called only one time to one of the many honeypot destination numbers. This explains the lower effectiveness of the blacklist shown in Figure 4.2. On the other hand, given the more ad hoc nature of the online user complaints collected in the COC dataset, it is not surprising to see a CBR reaching between 50-60%, when setting $\theta_c = 5$ for the COCNC blacklist, as shown in Figure 4.4. However, towards the end of our data collection period, we see a large drop in the CBR, including in the upper bound (i.e., $\theta_c = 1$) case. This means that the vast majority of numbers in the complaints collected every day after the drop were never seen before. After manual investigation, we found that many of the new complaints with never-before-seen source numbers seemed to be related to an IRS scam. Considering that the drop started in the weeks before the US tax filing deadline of April 15, it is possible that the drop is caused by a new large IRS scam campaign that relies heavily on caller ID spoofing.

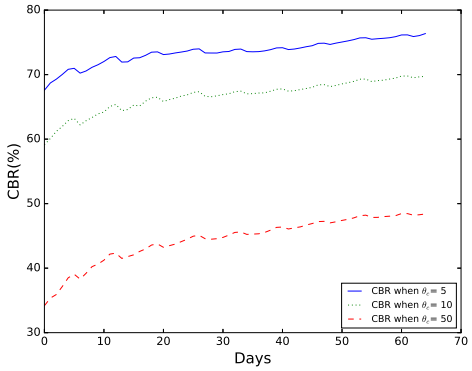


Figure 4.3: FTC complaints blocking rate

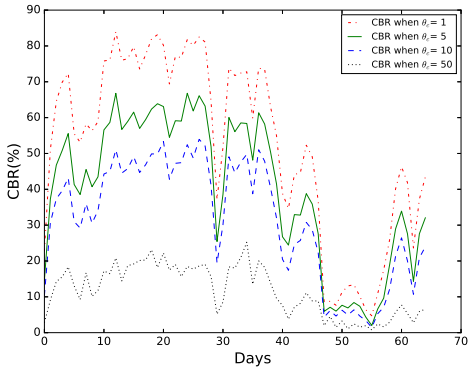


Figure 4.4: COCNC complaints blocking rate

4.6 Evaluating Context-Rich Blacklists

Context-rich blacklists (see Section subsection 3.3.3) tend to be much more conservative, compared to context-less blacklists, as clearly shown in Figure 4.1 (subsection 4.4.1). Unlike the context-less case, only numbers that can be attributed to one or more human-

identified spam campaigns are added to the blacklist. In addition, the HCT dataset only contains a small subset of the CDR data (i.e., only recorded calls). As expected, Figure 4.5 shows that the overall blocking rates for the HCT and COC blacklists are fairly low.

Consider that during training only a small fraction of the source phone numbers can be attributed to a distinguishable campaign. To see why this would be the case, let's consider the COC data source as an example. Figure 3.3 shows that a large number of source phone numbers are complained about only once or very few times and never again. This means that, in a particular day, many of the COC user complaints are related to numbers that were never seen before (and the will never be seen again). And because most user complaints contain only very short text, it is difficult to attribute these numbers to a campaign, and will therefore be excluded from the COC blacklist.

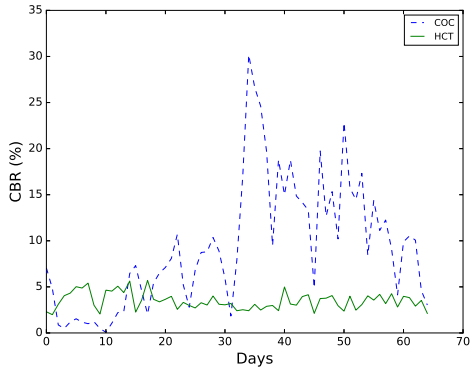
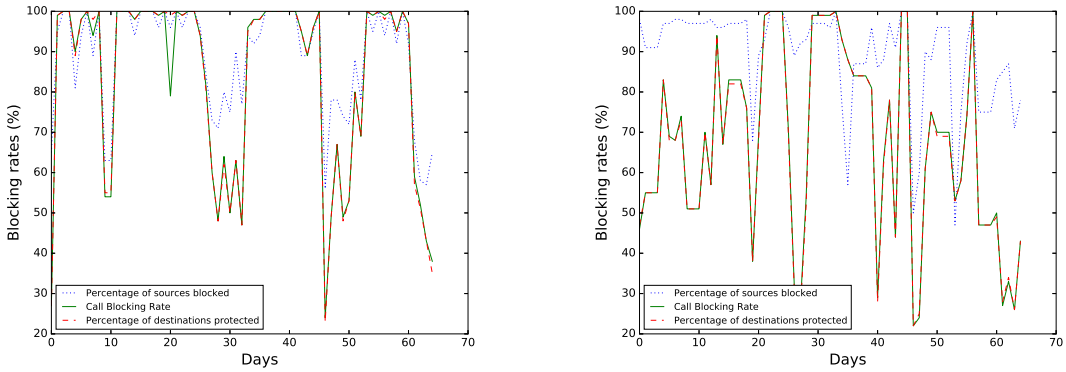


Figure 4.5: Overall CBR for COC and HCT

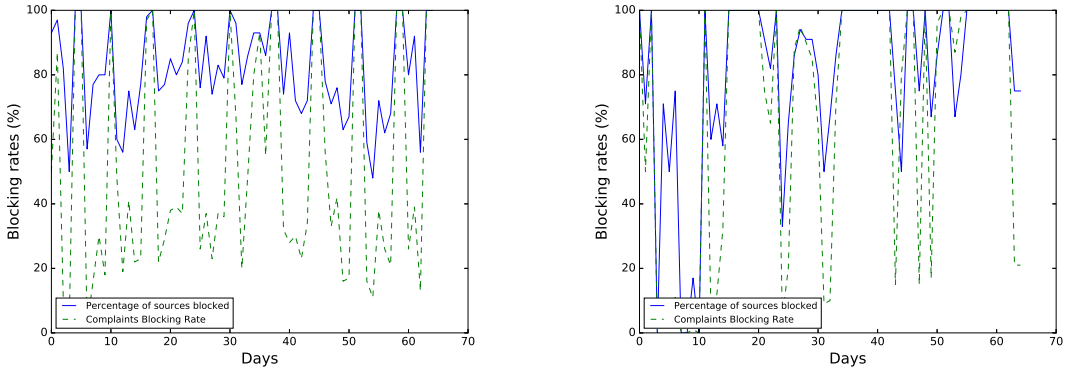
However, we should also consider how effective the context-rich blacklists are at blocking specific spam campaigns. To this end, below we analyze two representative campaigns discovered as part of the HCT blacklist learning (see subsubsection 3.3.3). Specifically, we explore the *Google Listings* and *Free Cruise* campaigns, and compute the CBR for calls from numbers that are assigned (via topic analysis) to these two campaigns, which are reported in Figure 4.6(a) and Figure 4.6(b). In addition, Figure 4.6(a) and Figure 4.6(b) also report the fraction of calling source numbers blocked and the fraction of destination numbers that are "protected" from the spam campaigns. We can notice that the CBR drops



(a) *Free Cruise* (b) *Google Listings*

Figure 4.6: Campaign call blocking rates over time.

significantly on some days, when many new source phone numbers appeared that were never seen before. However, the blacklist adapts fairly quickly to the new sources, by including these numbers at the next daily blacklist update, thus increasing the campaign CBR. In average, the CBRs for the two campaigns were 70% and 84%, respectively. These results suggest that while the spammers running these campaigns (especially the Free Cruise campaigns) do periodically churn through new phone numbers, they do not seem to employ caller ID spoofing as aggressively as one may think.



(a) *IRS* (b) *Tech Support*

Figure 4.7: Complaints blocking rate for top campaigns.

Figure 4.7(a) and Figure 4.7(b) show the CBR for two prevalent campaigns, the *IRS* and Tech support scams, that can be identified from the COC data. The figures show that

the source numbers used in these campaigns are more volatile than what we observed in the *Free Cruise* campaign in Figure 4.6, for example. This suggest that the spammers running these campaign may be more aggressively using caller ID spoofing, or frequently churning through new phone numbers. However, the average CBR is above 60% showing that the COC blacklist can still effectively block a meaningful fraction of calls belonging to these campaigns.

4.7 Measuring False Positives

In the previous subsections we have shown how effective blacklists are at blocking potential spam calls. Naturally, a high blocking rate should be paired with a low false positive rate, for a blacklist to be useful in practice. Ideally, to measure a blacklist's false positive rate we would need access to a large whitelist of legitimate phone numbers that have never engaged in spamming activities. Unfortunately, we are not aware of the existence of any such whitelist.

Because no ready-to-use phone whitelist is available, to estimate the false positive rate of our blacklists we proceeded as follows. We first built an instrumented browser (using Selenium WebDriver) capable of crawling the YellowPages directory [96], which lists the phone numbers of businesses around the US. The assumption is that the vast majority of businesses that advertise on YellowPages are legitimate entities unlikely to engage in phone spam activities.

Using our crawler, we gathered around 100,000 phone numbers listed across 15 different major US cities and 10 different business categories, including doctors, plumbers, insurance, restaurants etc. We then checked each of these numbers against our blacklists, and found that only 10 of them were present, yielding a false positive rate of only 0.01%. We further investigated the phone numbers that resulted in these false positives. We found that 7 of these phone numbers appeared in the FTC blacklist with 20 complaints per phone number on the average. The remaining 3 phone numbers appeared in the CDR blacklist

and on the average, they made 14 calls to 7 destinations. We also found that all of these 10 phone numbers have been complained about on 800notes.com for making unwanted or annoying calls.

According to YellowPages, the phone numbers which led to false positives belong to medical centers, plumbing businesses, locksmiths, auto repair shops and grocery stores. In 800notes.com reports, users mentioned that these numbers were involved in making robocalls about an insurance scam or the caller claimed to be an Amazon associate asking for money. Some complaints mentioned the calls came from annoying telemarketers and debt collectors. One possible explanation for this is that, while belonging to seemingly legitimate businesses, these numbers may have been spoofed by spammers as part of a phone spam campaign. If this is true, the very low false positive rate suggests that such spoofing is not common.

To assess the FP rate, we used the data described in subsection 3.2.5. Specifically, we used 100,000 benign phone numbers of businesses that were randomly chosen from YellowPages. While not complete, we believe this set of numbers is sufficiently large to yield a meaningful and reasonably accurate estimate of the FP rate.

4.8 Phone Abuse Campaigns

A natural question is whether the same phone numbers are used across different spam campaigns. In such cases, including a phone number on a blacklist due to abuse related to one scam could also protect users from other scams. Also, are the campaigns seen across multiple datasets or do they have higher visibility in a specific data source? We explore these questions by studying several prominent campaigns (derived from LSI topic modeling of our COC and HCT datasets). We found that the *Free Cruise* scam, shown in Figure 4.6, is seen in both the COC and HCT data sources, whereas the *Tech Support* and *IRS* scams shown in Figure 4.7 are largely seen only in the COC dataset and the *Google Listing* scam is seen in the HCT dataset. Figure 4.8 shows traffic over time for the top four

campaigns in HCT and COC datasets

We used the COC dataset to further explore various abuse campaigns. For example, we conducted a pairwise analysis of a number of additional campaigns, including *Home Security* and *Pay Day Loan* scams, and we found a considerable amount of overlap in source numbers involved in separate campaigns. For example, 231 of the 500 phone numbers used in the *Free Cruise* scam (see Figure 4.6) are also used in the *Pay Day Loan* scam (notice that both campaigns target typical consumers). Similarly, we have found around 90 phone numbers that were used for both IRS and Tech Support scams. While it is possible that different scammers may use caller ID spoofing and the same phone number can be spoofed in two unrelated scams, this is highly unlikely for two scams that independently spoof numbers roughly at random, given the size of the phone numbers space. Therefore, it is more plausible that the same spammers are responsible for multiple spam campaigns.

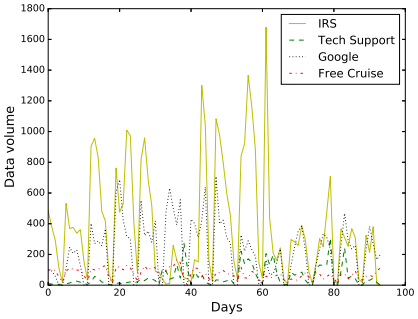


Figure 4.8: Traffic over time

4.9 Discussion and Limitations

The CBR rates shown in this chapter were computed using cumulative blacklists. In other words, the blacklists were updated daily by adding new phone numbers, but old phone numbers were never removed. One may infer that CBR rates computed in such a way can be overly optimistic, as old phone numbers may get reassigned to legitimate users. Therefore, retiring old numbers from the blacklist is a reasonable practice. To demonstrate the effect of removing older phone numbers from the blacklist, we have recomputed the

CBR rates by updating the blacklist in a non-cumulative way. In other words, we define a window of size n and remove any phone numbers that were not seen in the last n days. Figure 4.9 shows that the CBR rates of COCNC drop by about 1%-15% depending on the window sizes. We get similar results with FTC CBR rates.

Our results show that current telephony abuse can be mitigated with phone blacklists. However, if such blacklists are deployed widely, scammers can utilize a number of techniques to evade them. The ease with which source phone numbers can be spoofed makes such evasion easy. In fact, we are already witnessing that scammers spoof a number that has the same area code as the victim to increase the likelihood that the call will be picked up by a targeted victim. An analysis of recent CDR data shows that there has been a 20% rise in neighbor spoofing in 2017.

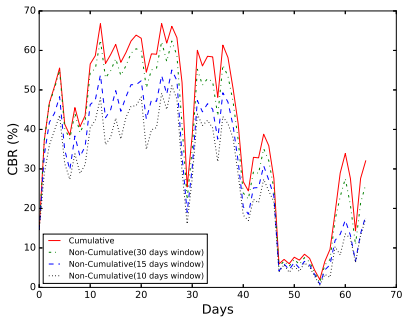


Figure 4.9: COCNC CBR rates with different window sizes

Although spoofing can make phone blacklists less effective, we believe that this will be more challenging for scammers in the future because of several recent initiatives [28]. For example, the Federal Communications Commission (FCC) has already proposed rules that allow carriers to block calls coming from unassigned or invalid phone numbers [97]. In absence of spoofing, the cost of acquiring a new spamming phone number will be non-negligible. Besides the cost of purchasing a number, the attacker risks that the carrier from which the numbers are acquired could block all numbers purchased by an entity that has been found to engage in any spamming activities.

Scammers can also evade the blacklists by making fewer calls from a source phone

number so the volume stays below a threshold such as θ_c used in the CDR blacklist. However, this will require that they utilize a larger pool of source phone numbers either by spoofing or by purchasing additional numbers. If spoofing can be made more difficult, evading the threshold will add cost to the phone scam operators. Our blacklists also rely on user provided data in the FTC and COC datasets. Scammers could target these data sources by injecting noise into them to reduce the effectiveness of blacklists. Also, a user may mistakenly provide incorrect information. For example, when a user receives multiple scam calls, they could incorrectly report the context of a call. Similarly, a more sophisticated attacker may check for liveness before providing call details in the HCT dataset. If such tactics become common in the future, the effectiveness of the blacklists could degrade unless techniques are employed to collect higher quality and more complete data.

4.10 Conclusions

Our results show that phone blacklists could block a significant fraction of unwanted/scam calls (over 55%). We also demonstrate that blacklists can be an effective defense against major phone abuse campaigns that have targeted consumers and businesses.

Currently phone blacklists can be effective against unwanted and scam calls but their effectiveness can suffer with increased level of caller ID spoofing. As spoofing has increased drastically recently, solely relying on blacklists is not enough to combat robocalls. We address caller ID spoofing in the next chapters.

CHAPTER 5
FIGHTING MASS ROBOCALLERS WITH A SMARTPHONE VIRTUAL ASSISTANT

5.1 Introduction

Mass robocalling, which targets millions of people with unwanted phone calls, is used to deliver voice spam and commit telephony abuse and fraud. Cheap mass robocalling , voice phishing [98] and caller ID spoofing [3] are some of the techniques that are being used by fraudsters in these attacks. Although phone blacklisting can be somewhat effective against scam calls, in the previous chapter we discuss how their effectiveness can be degraded with caller ID spoofing. Such spoofing is easy to achieve, and robocallers have resorted to tricks like neighbor spoofing (caller ID is similar to the targeted phone number) to overcome call blocking and to increase the likelihood that the targeted user will receive the call. To help reduce caller ID spoofing, both industry groups and regulatory bodies have explored stronger authentication for call sources. However, elimination of caller ID spoofing will not make all unwanted calls go away, as phone numbers can be cheaply acquired and used to overcome blacklists.

To detect unwanted robocalls and to provide the user with more meaningful call context when a phone rings, compared to only relying on the (spoofable) caller ID, we introduce RobocallGuard, a natural voice interaction model which is mediated by a Virtual Assistant (VA). The VA mimics a human call screener (e.g., a secretary) who picks up an incoming phone call and makes the user aware of the call only when it confirms that the call is not a robocall or other type of spam. When a call arrives, if the caller ID is not among the user's contact list, the VA transparently picks up the call and briefly interacts with the caller to determine if its source is a robocaller. Such interaction aims to be natural for legitimate

callers, while enabling the detection of robocall sources who indiscriminately target a large number of victims. Furthermore, such interaction with the VA enables learning the context of the call. Calls that are not detected as spam are passed on to the user, and the context extracted from the conversation between the VA and the caller is provided simultaneously, allowing the user to make an informed decision on whether the call is unwanted or legitimate.

Recently, a number of automated caller engagement systems that attempt to collect information about a call source have been proposed. To the best of our knowledge, no systematic usability and effectiveness studies have been reported of such caller engagement systems. Our goal is to explore an automated voice-based interaction approach that maintains both caller and callee user experience, eliminates user interruption and stops unwanted calls even in the presence of spoofed calls. We evaluate RobocallGuard's detection capabilities with a corpus of real robocalls and conduct a user study to evaluate its usability.

Although it may not be possible to stop all unwanted calls, we believe more trusted communication via the telephony channel can be supported by an automated call screening agent that can detect and block such calls without degrading user experience. The voice interaction model investigated in this chapter aims to help achieve this goal and we provide a proof-of-concept demonstration that it could be easily supported by current smartphones.

In summary we make the following contributions in this chapter.

- To the best of our knowledge, we are the first to evaluate a call screening virtual assistant that uses automated call handling and audio analysis to defend against robocalls and other types of spam calls, including those that evade blacklists with caller ID spoofing. In addition, transcription of call audio recorded by the virtual assistant is used to provide meaningful context about incoming calls to a user when the phone rings.

- Our virtual assistant aims to provide a mechanism that is similar, albeit much less "sophisticated", to having a human call screener who can pick up phone calls and

only forward to the user those calls which are likely wanted and record messages for the calls that are most likely unwanted. As a result, users are not annoyed with continuous ringing from unwanted calls.

- To demonstrate the ability of the virtual assistant to detect robocalls, we have developed a proof-of-concept smartphone app named *RobocallGuard*. To this end, we experimented with a corpus of 8,000 real robocalls collected by a large phone honeypot, and show that all of them can be detected and thus blocked.

- In addition, RobocallGuard allowed us to conduct an institutional review board (IRB)-approved user study to assess the usability of our virtual assistant. The results of this study demonstrate that the natural experience of a typical phone call is preserved for both callers and receivers, while benefiting from the ability to detect robocalls and other potentially unwanted calls.

5.2 System Design

5.2.1 System Overview

In this chapter, we propose RobocallGuard, a virtual assistant (VA) based solution that can help defend against unsolicited phone calls. We developed a smartphone app which can screen incoming calls without user interruption and intervention. The app hosts a VA, which works as a human secretary and receives incoming calls on behalf of the user. If the incoming call is from a whitelisted caller, the VA does not pick up the call and immediately notifies the user by ringing the phone. A whitelist can be defined by the user, and can include the user's contact list and other allowed caller IDs (such as a global whitelist which consists of public schools, hospitals etc.). On the other hand, if the call is from a blacklisted caller, the VA blocks it and does not let the phone ring. However, if the caller ID belongs to neither a whitelist nor a blacklist, the VA picks up the call without ringing the phone and initiates a conversation with the caller to decide whether this call should be brought to the

attention of the user. To make this decision, the VA presents the caller with a challenge which must be passed to reach the callee. The challenge can be thought of as an audio captcha which verifies the legitimacy of callers. To keep the caller experience natural, we experiment with a simple audio captcha, the name of the callee. Upon picking up the call, the VA asks the caller to state the name of the callee. If the caller says the correct name, the call is passed to the callee by ringing the phone. The transcript of the conversation between the VA and the caller is also shown on the phone screen to provide the callee with additional context. If the caller can not pass the above mentioned challenge, the VA blocks the call and notifies the user of the blocked call through a pop-up app notification. The VA also makes a decision if the call is from an unwanted human caller or a robocaller (discussed in detail in the later sections). Upon making this decision, the VA ends the call and stores the audio recording and transcript of the call for the user's convenience. Each audio recording and transcript is appropriately labelled (unwanted human caller or robocaller) by the VA. Since our proposed solution does not depend on the availability of a blacklist of known robocallers, it can be effective even in the presence of caller ID spoofing.

5.2.2 Threat Model

In-scope Threats

In this section we describe the scope of threats that our virtual assistant is designed to protect against.

Mass robocalls: Previous analysis shows that most of the robocall attacks that took place recently are mass calls. Attackers architect several campaigns such as tech support, IRS, free cruise and so on to reach a large number of phone users. Since the goal is to get a large coverage at minimal cost, attackers of such spam campaigns rarely target their victims individually. As a result, a simple audio challenge provided by the VA in the beginning of the call can filter out mass robocalls. Callers that cannot pass this challenge are not able to directly reach the callee.

Mass unwanted live calls from human: Our defense mechanism not only protects against robocallers, but also from unwanted human callers such as telemarketers and debt collectors who repeatedly try to reach people at wrong phone numbers. Unless, the caller knows the name of the callee, the callee is not interrupted by the call and is only notified asynchronously via a message that includes call context.

Spoofed Calls: The use of caller ID spoofing has increased significantly in the phone fraud eco-system. Neighbor spoofing is a common tool used by attackers these days. In our system, all incoming calls are picked up by the VA first, and audio analysis and speech recognition techniques are applied to prevent spoofed calls from interrupting the callee.

AI equipped attack: Attacks where the attacker is equipped with AI are not common in the phone fraud eco-system. However, with the availability of tools like Google Duplex [99], attackers can craft more sophisticated attacks where robocallers make a natural conversation with the other party and bypass our proposed defense mechanism. However, unless the attacker knows the name of the callee (which is not the case in mass robocalling campaigns), their call will not be passed to the user. The VA might mislabel an AI equipped robocaller as an *unwanted* human caller, but will still be able to stop the unwanted call from reaching the user.

Out of scope Threats

Our VA does not protect against the following types of attacks. The attacks discussed below are currently not common, but may emerge in the future in an attempt to defeat intelligent phone call defense tools such as the one we propose in this chapter.

Targeted attack: Our VA only protects against mass spam/scam calls. If the attackers obtain the callee's name associated with a smart phone number through leaked private data, they can evade the VA by saying the correct name. Currently, such targeted attacks are rare, but the increase of leaked private information may pose such threat in the future. We discuss this in detail in section 5.5 and explore a defense in Chapter 6.

Landline calls: RobocallGuard only protects callees when they use a smartphone. Hence, malicious actors making landline calls using public directories are out of scope.

Possible Evasions

A robocaller might try to bypass our proposed defense mechanism in the following ways.

Common name attacks: A common name attack is where a robocaller plays a prerecorded message of carefully chosen common names to bypass the VA. Since we evaluated the VA where the challenge is the callee's name, such an attack could evade the VA if the user has an identical or similar name to any of the common names used by the attacker. However, the challenge can be made more difficult for the attacker by requiring both first and last names.

Master key attack: Another possible evasion technique is crafting a keyword that can evade a large set of names. This keyword works as a master key that might fool the VA to accept the crafted keyword as the correct name.

5.2.3 Design Goals

In this section, we state the design goals that are required from a defense system designed to combat unwanted calls. Such a system needs to perform content analysis since relying only on caller ID is not sufficient to stop unwanted calls.

- *Add an extra layer of security between the caller and the recipient of the call.* This facilitates that all calls from *unknown* phone numbers (i.e., phone numbers not stored in the recipient's contact list) are passed through the VA, which filters out robocalls and unwanted human calls. The motivation behind this design goal is to add a challenge to the caller before the callee's phone rings and interrupts him/her. The challenge should be easy and natural enough for a legitimate caller to pass, but harder for mass robocallers to pass.

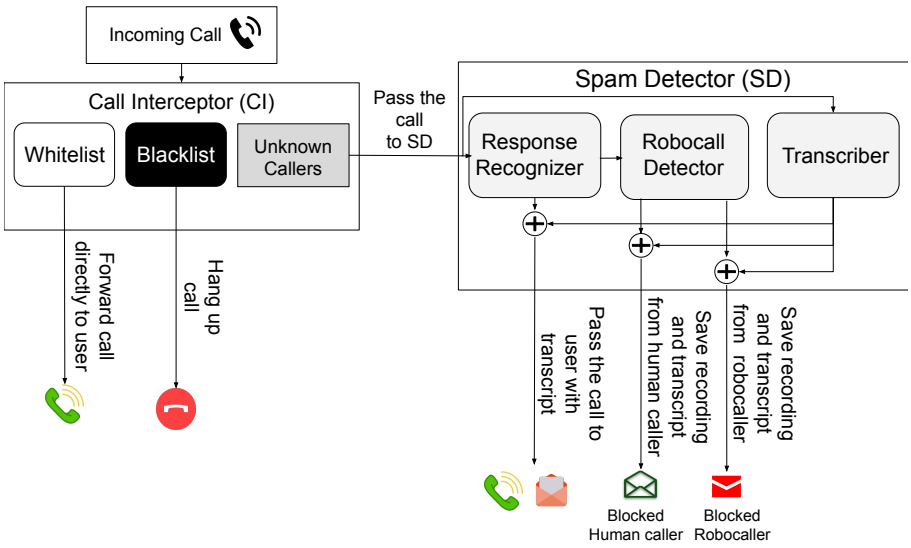


Figure 5.1: System Architecture

- *Provide additional information to the recipient about the content of the call*, so that they can make an informed decision to pick up the call. Such information should be extracted from audio received from the caller before the call is picked up by the callee.

- *Preserve user experience* in regards of latency and accuracy. The VA should not make the phone call experience too unnatural for the caller or callee.

- *Ensure privacy* when an incoming call is handled by the VA. In other words, the VA should run locally and the call is not transferred elsewhere. This ensures that the conversation between the caller and the VA is protected.

In our implementation of the VA we are able to achieve the first three goals. We also believe that ensuring privacy is feasible and we discuss it in detail in the later sections.

5.2.4 User Workflow

We consider a scenario in which smartphone users install RobocallGuard. All incoming calls are sent to the VA. Hence, the phone does not immediately ring and notify the callee of the incoming call (Figure 5.1). The VA makes a preliminary decision based on the

caller ID of the incoming call. Whitelisted callers are immediately passed to the callee and blacklisted caller are blocked. The following steps are followed to handle calls from neither blacklisted or whitelisted callers.

1. The VA picks up the call, greets the caller and asks for the name of the person the caller is trying to reach. This enables the VA to check if the caller knows the callee, and if it does not, it is likely that this call is unwanted.

2. If the caller says the correct name, the VA detects it and passes the call immediately to the callee by notifying them of the incoming call. The correct name is set by the user when they install the app. The VA also provides the transcript of the conversation it just had with the caller to the user. While passing the call, the VA asks the name of the caller as well and provides it to the callee along with the transcript.

3. After a predetermined time t_1 has elapsed from the initial greeting and the caller has not said the correct name, the VA asks for the name again and looks for caller interruption while playing this message. The intuition behind this is to differentiate robocallers from actual human callers. Current robocallers typically play a pre-recorded message and do not stop to make a conversation with the callee. Therefore, the callers who are not interrupted by the VA are labeled as robocallers. On the other hand, if the caller is silent during most of the time the VA is talking, we label them as potential unwanted human callers.

4. After t_2 seconds have passed from the initial greeting and the caller has not said the correct name, the VA hangs up the call and saves the entire audio recording of the conversation it had with the caller. The VA also saves a transcript of the audio. The audio recording and transcript is labeled as robocaller or human caller according to the previous step. Finally, the callee is notified of this blocked call and provided with the audio recording and transcripts. This allows the user to make a decision if they want to call back the caller or not.

5.2.5 RobocallGuard Architecture

In this section, we describe our system architecture which is independent of the implementation environment. The underlying architecture of our virtual assistant is designed to fulfill the goals mentioned in subsection 5.2.3. To achieve our first goal of enhanced security, the modules, Call Interceptor and Spam Detector, act as a middle layer. They collect and analyze additional call information to enhance security against unwanted calls. The communication flow between the system components ensure that user experience is preserved. To ensure privacy, our envisioned system handles all calls locally. This could be achieved by implementing the VA as part of the default phone app, for example. In contrast, Robokiller [31] routes all incoming calls to a central server, thus exposing possibly sensitive audio to a third-party. Figure 5.1 depicts the main components that make up our VA.

We envision our system to be embedded in the Phone app of a smartphone. The call screening feature provided by Google Pixel [100] phones demonstrates the feasibility of embedding a VA with the phone app to intercept and examine voice from incoming phone calls. However, due to certain OS enforced restrictions, we implement a proof-of-concept prototype instead of embedding the VA with the Phone app. Such limitations and the choices we make to overcome them are discussed in section 5.3. In the following, we describe each component of the system architecture.

Call Interceptor

All incoming calls are passed to the Call Interceptor(CI) module. The main function of the CI module is interception of a call to acquire the incoming audio stream, and injection of recorded voice messages by the VA into the outgoing audio stream. The CI makes an initial decision based on the caller ID. All calls from whitelisted phone numbers are passed to the user (callee in this case) without further processing. A user has total control of the whitelist on his or her phone and can decide phone numbers from which calls should be

passed to them directly without intervention from the VA. All calls from blacklisted phone numbers are dropped and stopped from reaching the user. The blacklist is predefined as well; however, it is designed to be dynamic to include newly appearing malicious phone numbers [49]. Phone numbers which are not present in the whitelist or blacklist are labeled as "Unknown Callers". Audio stream from all unknown callers are passed to the Spam Detector module for further analysis.

Spam Detector

This module analyzes the audio coming from an unknown caller to make a decision about the nature of the incoming call.

Response Recognizer Incoming audio stream from calls originating from unknown callers are passed from the Call Interceptor to the Response Recognizer(RR) module. The RR module decides whether to pass the incoming call to the callee. If the call is considered unwanted, it is handled by the VA and not passed to the user. However, since the notion of an unwanted call is different for each user, it is difficult to define an unwanted call. Hence, we take a conservative approach: if the caller knows the name of the callee, we label that call as *wanted*; conversely, when the caller does not know the name of the callee, that call is labelled as *unwanted*. The intuition behind this approach is that phone calls from a person who knows the callee are less likely to be unwanted.

The user is allowed to set the name(s) that should be accepted as correct by the VA. We refer to the name(s) set by the user during the installation of the app as *correct name(s)*. Users may set multiple *correct names* as a valid recipient of phone calls coming to their device. After the installation, during a future phone call, the caller is asked who they are trying to reach. After the call has been picked up, we set a limit of 35 seconds (value of time limit t_2) to allow the caller to say a *correct name(s)*. The value of t_2 is set empirically, keeping in mind that the VA speaks for 15 seconds during the 35 second time limit, hence

the the remaining time period should be enough for the caller to provide meaningful context. Moreover, t_2 can be tuned to match user experience and context details. If the caller says any of the *correct names* at any point during this 35 second period, the RR module recognizes it and passes the call to the user along with the transcript of the conversation between the VA and the caller. However, if the caller does not say any of the *correct names*, the call is deemed as unwanted and not passed to the user.

The backbone of the RR module is a keyword spotting algorithm which can detect the right keyword. In our scenario, the correct name(s) of the callee is the keyword. There has been a lot of research on keyword spotting algorithms which are used in many commercially available products such as Amazon Alexa, Okay Google in Google products etc. Hence, we explored existing systems that can effectively detect a keyword. However, for such a system to be usable in our VA, high accuracy with limited training examples, an open source toolkit and a light enough system to run on a mobile device is required.

Since the users set the *correct name* by making audio recordings of them pronouncing the names, it is not feasible to collect a large number of audio samples from the users. In other words, the keyword spotting algorithm will have access to only a few recordings of each name. However, the system should have a high true positive rate, and a low false negative rate. Snowboy [101], CMU Pocketsphinx [102], Honk [103] are all open source keyword spotting toolkits that are light enough to run on a mobile device. Based on our experiments, we found CMU Pocketsphinx has lower accuracy than Snowboy, when trained with names. Honk requires a larger number of audio samples to train a keyword. On the contrary, Snowboy requires only 3 audio recordings to train a keyword. Snowboy also supports multiple keyword models, thus multiple names can be set as keywords. Hence, we chose Snowboy to recognize the name. Because Snowboy does not connect to the Internet, it can ensure privacy, which is one of our design goals. We treat Snowboy as a blackbox, which when provided with 3 audio samples, creates a model to detect the keyword. We embedded the downloaded trained model with the VA to recognize the *correct name(s)*.

Robocall Detector The objective of this module is to determine whether the caller is an actual human or a robocaller. When a caller cannot say the correct name, their call is handled by the Robocall Detector(RD) module.

As mentioned in the workflow subsection 5.2.4, the RD is activated after t_1 seconds. We set t_1 to 20 seconds as we want to give the caller enough time to say the correct name. The analysis of the data we collected from our user study shows that the initial 20 seconds is a reasonable time for the callers to say the correct name even if they have to repeat the name. After 20 seconds have passed from the initial greeting and the caller has not said the correct name, the VA plays an audio message to interrupt the caller. Let the duration of this audio played by the VA be t_3 seconds (we set t_3 to 5 seconds in our experiment). During these t_3 seconds, the RD module checks if there is silence from the caller's side. We use Voice Activity Detection (VAD) [104] to determine if the audio coming from the caller's side contains voice or silence. If the caller is silent for at least t_3/2 seconds while the VA is playing the audio message, we label the caller as an actual person. On the contrary, if the caller is silent for less than t_3/2 seconds, it is labelled as a robocaller. Determination of the type of the caller (human or robocaller) provides additional information to the user about the call. Upon determining the type of the caller, the VA allows the caller 10 more seconds as a margin of error to say the correct name before hanging up the call. The audio recording and transcripts of the entire conversation with the caller is saved locally at the device for the user to preview later. The associated label (human or robocaller) is used to determine in which folder the audio and transcript is stored.

Transcriber The transcriber component transcribes the entire conversation between the VA and the caller to provide additional context to the user. The VA stores the audio recording and the transcription of that audio recording locally; and notifies the user of these two files after it has handled the incoming call. This helps the user to access the content of the call without picking up and engaging in the phone call. Calls that are passed to the

user by the VA after deeming them as "wanted", are also provided with a transcript of the conversation between the caller and the VA that took place before the call was forwarded to the user. When the user is notified of a such a call by ringing the phone, the transcript is shown on the screen for additional context. Calls which are deemed "unwanted", both from human callers and robocallers, are not passed to the user and hung up by the VA.

The VA does not engage in a conversation with callers that are whitelisted and passes the calls directly to the user. Therefore, transcripts are not provided for such calls. All other callers are greeted by the VA and hence transcript is made available to the user to understand the content of the calls.

There are many software libraries and APIs available for transcription. We have chosen Google Cloud Speech API [105] because it has a very high accuracy and transcription can be performed from a mobile device, unlike Kaldi [80] and Mozilla deep speech [106]. Ideally, the transcription should be conducted locally in the device and no server should be involved. Android provides the means to perform transcription locally through *RecognizerIntent* and *SpeechRecognizer* class. However it imposes the restriction of the input channel. *RecognizerIntent* and *SpeechRecognizer* class always uses audio from the microphone as an input for transcription. As discussed in section 5.3, audio from the caller in our implementation comes though the VoIP channel instead of the microphone. Therefore, in our proof-of-concept prototype, we do not perform transcription locally. Instead, we send the stored audio recording of the conversation between the VA and the caller to Google Cloud and a corresponding transcript is returned. However, when RobocallGuard is deployed in real life, transcription can be done locally in the device, thus privacy can be ensured.

5.3 Implementation

In this section, we discuss some important details of our implementation of the VA. We implemented a prototype of our app using Java on Android. We envision our VA to be embedded with the Phone app where the VA handles all incoming calls locally without

having to stream audio to an external server. The recent release of call screening feature for Android phones further supports the feasibility of such a system. However current Android system restrictions do not allow embedding a system that can inject voice messages in the outgoing audio stream without OS modifications. Hence, in the following, we describe and explain the choices we made in the implementation of our proof-of-concept prototype.

5.3.1 Motivation behind VoIP

The workflow of the VA starts with an incoming call being passed to the CI module. As discussed earlier, the CI module captures the audio stream from caller side and takes full control of the stream so that the system is capable of analyzing the voice data, as well as locally recording the caller's audio. Moreover, the CI also injects audio into the phone call to communicate with the caller on behalf of call recipient, while the recipient has no awareness of the incoming call during the time the VA is interacting with the caller. To perform the first task, we could implement a customized phone call app by making modifications to phone call service codebase provided by Android (i.e. Implementing customized Android.telecom.InCallService). However, it is strictly constrained for a common developer to satisfy all the requirements for injecting audio to an ongoing phone call. For the sake of security and privacy, Android does not allow injecting sound files in the conversation during a phone call [107], which means that no such API is provided by the Android system that could pre-process or replace the microphone as an input audio stream during a phone call.

Taking all these into account, we decided to implement a VoIP(Voice over Internet Protocol) application to conduct our user study experiment. With an VoIP phone call application, we are able to get full access to voice streams on both sides of a phone call. Furthermore, it is possible to inject audio at any appropriate moment in the conversation during a phone call with a VoIP app.

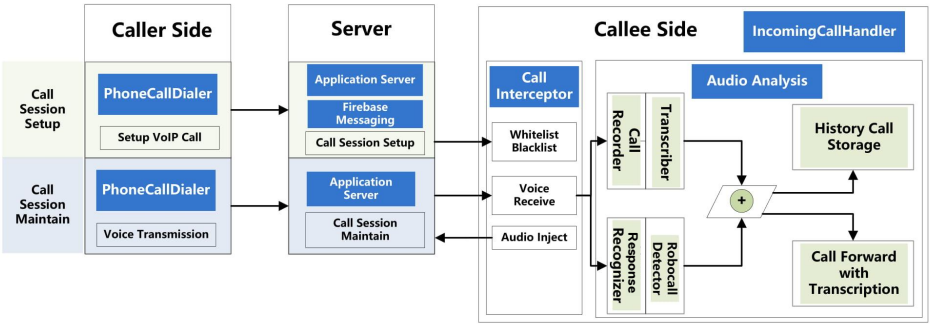


Figure 5.2: VoIP Architecture

5.3.2 VoIP application architecture

Our VoIP application consists of two major parts, namely client-side and server-side systems. For the sake of convenience, we implemented the system so the two parts work over TCP network connections by using a simplified and customized session initiation and keep protocol. Figure 5.2 demonstrates the architecture of our VoIP application.

On the server side, an application server (Server) working along with Google Firebase Cloud Messaging Platform (FCM) [108], handles call session setup and maintaintenance. On the client-side, we split the core of our call implementation into two parts, one called "PhoneCallDialer (PCD)" and the other called "IncomingCallHandler (ICH)". PCD is mainly a dial service, which provides a dialing panel for the user to make VoIP phone calls. At call session setup stage, PCD initiates a connection request and builds a call session between caller and callee with the help of the Server. At call session maintain stage, PCD streams and transmits caller's voice to callee through the channel maintained by Server.

ICH consists of two functional modules, which implement callee-side handling of incoming calls. The CI module automatically answers an incoming call without the callee's awareness during call session setup stage and makes the initial decision to forward calls from whitelisted numbers or deny calls from blacklisted numbers. For unknown callers, the call is passed to the Spam Detector (SD) module and corresponding pre-recorded audio

tracks are injected into the call to allow the VA to communicate with the caller.

We implemented and used the system described above to conduct a user study and evaluate the efficacy of the VA. The main purpose of developing a VoIP prototype is to assess the usability of the VA and its effectiveness in detecting robocalls and other unwanted calls. Although we developed a VoIP prototype, it is possible to embed our VA in the Phone app.

5.4 RobocallGuard Evaluation

In this section, we report the results of experiments we conducted to measure the accuracy of decisions made by our VA for incoming calls, and discuss a user study that was conducted to evaluate the usability of our prototype.

5.4.1 Usability Study

Our VA is designed to provide the convenience of a human assistant while detecting unwanted calls. It also provides context for calls, which helps the callee decide if a call needs his/her attention. To explore the usability of such a system, we conducted an Institutional Review Board (IRB) approved user study. In the following, we first describe the study setup, its participants and then discuss the results.

Study Setup

Our study participants consists of 21 users who were sampled from a population of college students. Most of the participants can be described as tech-savvy. All participants were required to be fluent in English and be familiar with using smartphones. Each experiment was conducted with two Android devices, a Samsung Galaxy S9 plus and a Samsung Galaxy Tab A, running Android 8.0 and 7.0 respectively. Both of the devices had RobocallGuard app installed. During the user study, all phone calls were made using the Tab and received using the smartphone. The setup of the experiment is as follows. We briefed the participants about the experiment process and explained the purpose of the VA. We asked

the users to perform a list of tasks: making calls, receiving calls, checking the contents of blocked calls and answering multiple choice questions regarding their experience of using RobocallGuard. Participants were assigned the role of a caller and a callee one at a time and were asked to make/receive a call. When participants were assigned the role of a caller, they made 4 calls. Such a call took at most one minute. When participants were assigned the role of a callee, they received 5 calls. After making/receiving each call, the participants had to answer multiple choice questions regarding their experience. At the end of the user study, each user was asked 6 generic questions about their overall experience with the app.

User Actions

In this section, we describe each task the participants performed during the user study in detail. Each experiment involved a pair of users (user A and user B), one caller and one callee, performing the tasks. Once user A has completed all the tasks assigned to the caller, they are assigned the role of a callee and vice versa. The experiment starts with user A acting as the caller and making calls to the callee, user B.

We performed two experiments within each experiment. During the first experiment, we provided the caller with the appropiate response to the challenge i.e. the correct name. Hence, the caller should be able to reach the callee. Conversely, in the second experiment, the caller is either given an incorrect name or no name at all. Therefore, the VA would not allow them to reach the callee. The participants had no idea about what the correct name was. Furthermore, there are two scenarios in each sub-experiment; one where the caller is provided with a script to read from when making a conversation with the VA, and one where the caller is given a topic to talk about, instead of a script, while interacting with the virtual assistant (e.g calling a friend to make movie plans.) When a user is assigned the role of a callee, with each forwarded call they are given the choice to either pick up or decline the call. They are advised to use the caller-VA interaction transcript provided by the app to make this decision. Once they pick up the call they can start a normal conversation with

the caller. During our user study, we preset the correct name to be *Taylor* instead of having each user set a name. We make this choice because the purpose of the user study is to get insights about call experience in the presence of a VA, rather than testing the accuracy of the keyword spotting algorithm. We ask the callee to impersonate Taylor when making the decision to answer an incoming call. Furthermore, the callee has no advance knowledge of the content of the call. Following is the detailed description of the 4 calls made by the caller during the experiment.

Experiment 1: In this case, the correct name "Taylor" is provided to the caller and the callee is asked to impersonate Taylor.

Scenario A: When asked the name of the callee, the caller is instructed to read the following script, "Hello, can you please forward my call to Taylor?". In this scenario the VA allows the caller to reach the callee.

Scenario B: In this scenario, the caller is not given a script. They are instructed to make a call to their friend Taylor to make movie plans. They are advised to include the name Taylor in their conversation.

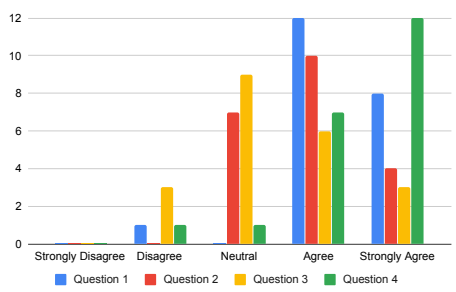
While forwarding the call to the callee, the VA asks the caller to state their name. We advised our participants to say a fake name to protect their privacy.

Experiment 2: In this case, an incorrect or no name is provided to the caller and similar to Experiment 1, the callee is asked to impersonate Taylor, the correct name being set as Taylor.

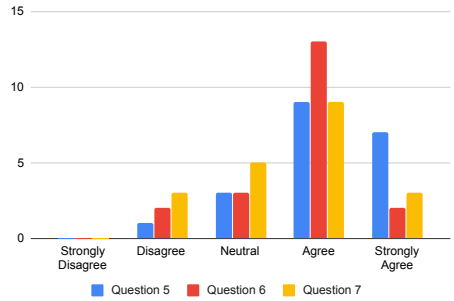
Scenario A: In this scenario the caller is asked to read the following script, "Hello, can you please forward my call to Robert? We met at a seminar today." It should be noted that Robert is the incorrect name here, hence the VA blocks this call.

Scenario B: In this scenario, the caller is not given any script or name. They are instructed to make a call to an office trying to sell a computer. Since the caller does not say the correct name during this call, the VA blocks this call as well.

As a result the callee gets a notification of the blocked call along with the transcript of



(a) Caller and Callee Response



(b) User Generic Response

Figure 5.3: User Results

the call. After making each call, the caller is required to answer survey questions that focus on the ease of interacting with the VA, the quality of the transition from the VA to the callee and the delay experienced before the callee responded. As a callee, a participant received the aforementioned 4 calls. The VA passes the first two calls to the callee since the correct name was said by the caller. On the other hand, the VA blocked the last two calls, notified the callee of the blocked calls and provided her with the transcript and audio recording of the conversation. In addition to receiving the 4 calls made by the caller, the participant received a robocall made by us. The VA blocks this call, labels it as a robocall and notifies the user. After each call (both blocked and passed), the callee is required to answer a number of survey questions regarding the usefulness of the transcript, the interaction with the caller, and the reason behind their decision to pick up or not pick up the call.

User Study Results

In this section we present the results obtained from the user responses to the survey questions. We present user responses to the following seven questions in Figure 5.3. Questions 1-2, 3-4, 5-7 investigate caller, callee and overall user experience respectively.

- Question 1: It was easy to interact with the VA.
- Question 2: The delay you experienced before the other person responded to the call is acceptable.

- Question 3: The transcript was able to provide sufficient information to infer the topic of the incoming calls.
- Question 4: The transcript was able to provide sufficient information about the content of the blocked calls.
- Question 5: I found the app beneficial to me as it provides prior knowledge about the incoming calls.
- Question 6: I think I would like to use an app equipped with a VA frequently.
- Question 7: I felt comfortable with the VA intervening in the phone calls.

Caller Experience: Most of the users who acted as the caller reported that it was easy to interact with the VA. Figure 5.3a shows the distribution of the responses to a question about the ease of interaction with the VA when the caller said the correct name. When the caller was given an incorrect or no name (in case of sub-experiment 2), 4 out of 21 users disagreed that it was easy to interact with the VA. This is understandable since the user had no knowledge of what the correct name is, they became frustrated when the VA did not pass their call to the callee, which is what may happen when a telemarketer calls. Figure 5.3a also shows the user responses regarding the delay experienced by the caller before the callee could be reached. Most users agreed or strongly agreed that the delay was acceptable. Moreover, all but one caller reported that they were satisfied with the transition from the VA to the callee. Hence, it can be concluded that a call experience for callers was not degraded as a result of having the VA acting as an intermediary.

Callee Experience: While acting as the callee, when a call is passed to the user, they may either answer or decline the call. Most of the users who picked up the call said that they made this decision because the transcript suggested this to be a non-spam call. Only 4 users reported that they answered the call because they pick up all incoming phone calls. All but one callee said that the interaction with the caller felt normal/natural after they picked

up the call. Figure 5.3a shows how the callees felt about the transcript of the incoming calls. Only 3 users reported that the transcript was not able to provide with sufficient information about the incoming calls. The reason behind this is that some callers only said the correct name "Taylor" and nothing else after starting a conversation with the VA. Thus, the transcript consists of only the correct name and no other information. Since it is not possible to control what the caller says, the quality of the transcript cannot be guaranteed in all cases. However, since the caller is also required to say their own name before a call is forwarded, it provides further information about the call to the callee. The current version of our prototype does not ask the caller to state the purpose of the call. We plan to add this feature in our future version, which we expect will augment the content of the transcript. Figure 5.3a also shows callee responses about the transcript of the blocked calls. All callees reported that they were notified in a timely manner of a new blocked call.

Overall Experience: At the end of the user study, each user was asked 6 generic questions. According to Figure 5.3b, most users mentioned that they found the app beneficial to them as it provides prior knowledge about incoming calls. Moreover, most users said that they would like to use an app equipped with a virtual assistant frequently. In addition to these questions, all but two users said that the app was easy to use. Only 3 users reported that they needed to learn a lot of things before they could get going with this app and only two users reported that they could not easily navigate through the app. Moreover, all but 3 users reported that they were comfortable with the VA intervening in their phone calls before the calls are forwarded to the callee. Based on these results, we can conclude that the the VA does not negatively impact the overall call experience and lack of usability will likely not be the reason for impeding its adoption. Since we recruited 21 users for this study, the accuracy/quality of our conclusion from this study lies between 80% to 95% (between 10 and 20 users) according to [109].

5.4.2 Correctness of RobocallGuard

In this section, we evaluate the effectiveness of RobocallGuard based on its ability to forward legitimate calls and block and label unwanted calls. For each incoming call, the VA has to make a decision whether to forward the call to the user or not. Besides, it needs to label each unwanted caller as human or robocaller.

Interaction with Human Callers

We first measure the effectiveness of RobocallGuard when the caller is a human. We categorize all human callers into two catagories: legitimate callers and unwanted callers. We define a caller to be legitimate if they know the correct name of the callee. On the contrary when the caller fails to pass the challenge, we define them as unwanted callers. To measure RobocallGuard's effectiveness when callers are human, we used the data collected during the user study. We have discussed usability insights in our user study results in subsubsection 5.4.1; here we present statistics on RobocallGuard's accuracy.

Legitimate callers: As discussed earlier, during the user study, we performed four experiments with each of the 21 callers. In the first two experiments the callers were given the correct name of the callee. Our analysis shows that when callers passed the challenge, Robocall Guard detected all of them as legitimate callers and forwarded their calls to the callee. In only two cases, the callers had to repeat the correct name, and after the second time the callers had said the correct name, RobocallGuard forwarded the calls to the callee. This was mostly due to snowboy, and advancements in keyword spotting algorithms will further reduce such false negatives.

Unwanted Callers: During the user study we conducted, in the third experiment the callers were given an incorrect name and in the last one the callers were not given any name at all. Therefore, in these cases, the callers in our user study played the role of an unwanted caller in a way similar to a telemarketer, and made calls to the callee. The VA picked up the calls and was able to prevent every incoming unwanted call from directly reaching the

callee.

Interaction with Robocallers

In this experiment we measure the fraction of robocalls stopped and correctly labelled by the VA. We use a dataset of phone call records collected at a large phone honeypot provided by a commercial robocall blocking company. This dataset contains 8081 calls (which had an average duration of 32.3 seconds) coming into a telephony honeypot during April 23, 2018 and May 6, 2018 [1]. It records the source phone number that made the call, the time of the call, the audio recording of the call and the transcript of the audio recording. Since it is a telephony honeypot, it contains some misdialed calls along with robocalls.

Robocall Detection: To filter out misdialed calls, we use the approach previously discussed in Chapter 3 to extract important topics from the transcripts of calls in our robocall dataset. Using LSI topic modeling, we extract 60 topics from our corpus of transcripts where each topic represents a spam campaign. We construct a similarity matrix by computing the cosine similarity between each transcript. We then convert the similarity matrix into a distance matrix by inverting the elements of the similarity matrix. We performed DBSCAN clustering [110] on the distance matrix. DBSCAN is one of the most common clustering algorithms which given a set of points, groups together points that are closely packed together, marking as outliers points that lie alone in low-density regions. At the end of this, 79 clusters were created where each cluster represents groups of highly similar transcripts of robocalls. Since, the honeypot by nature contains unwanted calls, the clustering technique acts as a sieve that filters out all non-spam calls as outliers. Moreover, it allowed us to take one representative robocall from each cluster and use the audio recording to make a call to RobocallGuard. Upon making the calls, RobocallGuard correctly detected 100% of all robocalls as unwanted and stopped them from ringing the user's phone. Theoretically, a false negative occurs when RobocallGuard forwards an unwanted

[1] Although this dataset is not recent, robocaller behavior has not significantly changed.

call i.e. recognizes an incorrect response to the challenge as a correct one. A false positive occurs when RobocallGuard blocks a wanted call i.e. fails to detect the correct response to the challenge. With stricter challenges and fast advances in AI, we expect the correctness of RobocallGuard to increase.

Robocall Labeling: Once unwanted calls are detected, RobocallGuard determines if the caller sounds like a human or a robocaller. From the previous experiment, we noticed that the robocalls with a duration of less than 20 seconds were being inaccurately labeled as unwanted human callers. The reason lies in the design of RobocallGuard. RobocallGurad interrupts a caller at the 20th second and determines if they were interrupted or not by looking for voice activity from the caller side while the VA is speaking. The robocalls that are less than 20 seconds are found to be silent when the VA interrupts and are mislabeled as human. To solve this problem, we analyzed the contents of the short length robocalls. Since the robocallers are trying to financially profit from their victims, the content of the short robocalls must serve a purpose. From analysis of the robocall recordings in the honeypot, we discovered that 86% of the short robocallers ask the callee to press or enter a digit in the phone keypad. Hence we take a further step to identify the short robocalls. If the transcript of a call contains the keywords ”press” or ”enter”, we label it as a robocall. It is unlikely that a legitimate human caller will say these words while interacting with the VA. With this added step, our results show that RobocallGuard is able label 97.8% of all robocalls correctly.

5.4.3 Comparison with Call Blocking Apps

There are several commercial applications available in the app stores that aim at blocking robocalls. Most of these apps (such as Youmail, Hiya, Nomorobo, etc.) rely on phone blacklists. As discussed earlier, spoofed calls, which are common, cannot be blocked with this approach. Currently, only Robokiller claims to perform call content analysis, in addition to using phone blacklists, to block unwanted calls. We performed a small scale

experiment to compare Robokiller with RobocallGuard. To conduct this experiment, we installed Robokiller on a Samsung Galaxy S9 Plus Android device. We then used a Twilio [111] phone number to make 10 robocalls to the device where Robokiller is installed. We chose a random sample of 10 robocall messages. Since the phone number we used is a Twilio verified phone number, it is not a blacklisted phone number. Therefore, a defense system that relies only on phone blacklists will not be able to block these robocalls. Only a system that performs audio content analysis will be able to detect these robocalls. However, we noticed that Robokiller was not able to block any of the robocalls made from the Twilio phone number and let the calls pass to the user without any spam call warning. In contrast, RobocallGuard was able to block all of these 10 robocalls. Therefore, it can be inferred that the available Robokiller app seems to rely mostly on phone blacklists rather than call content analysis. To explore this further, we downloaded FTC user complaint reports from August 3-5, 2019 and extracted 10 phone numbers that had most complaints. We then spoofed each of these 10 caller IDs using SpoofCard [112] to make phone calls to Robokiller. Since we conducted this experiment on September 10, it can be expected that the top 10 callers from a month old FTC dataset will be in blacklists used by commercial apps. In addition, we performed reverse look up on each of these 10 phone numbers and found 7 of them to be labeled as "scam or fraud". Upon making the spoofed phone calls, we found that Robokiller was able to block calls from 9 of the 10 caller IDs. This shows that Robokiller is able to block most of the incoming calls from blacklisted caller IDs.

In our next experiment, we downloaded FTC user complaint reports from September 9, 2019 and extracted the 10 phone numbers that were least complained about. We spoofed each of these 10 caller IDs to make phone calls to Robokiller. Since this dataset contained complaints from the previous day, it can be assumed that these phone numbers are not present in blacklists. Upon making the spoofed calls, we found out that Robokiller let calls from 8 of the 10 caller IDs pass. Although this is a small scale experiment, it provides evidence that Robokiller appears to rely more on phone blacklists and not the content of

the call. Hence, unlike our proposed VA, robocalls with spoofed or previously unseen caller IDs can evade Robokiller.

5.5 Discussion

RobocallGuard allowed us to test the usability of a VA and its effectiveness in stopping current robocalls. The survey responses from the user study we conducted show that both callers and callees are comfortable with the change in the call experience due to the VA and found RobocallGuard beneficial. Moreover, [113] has shown that users need a app which handles spam calls without making the phone inoperable. The fact that RobocallGuard filters out spam call without user intervention and without making the phone inoperable adds desired convenience to the users. Much of the current voice abuse over telephony is perpetrated by mass robocallers who indiscriminately call a large number of potential victims. They use techniques like neighbor spoofing to increase the likelihood that their calls are picked up. Our results show that RobocallGuard can be effective against such mass robocallers.

We understand that the callee's name might seem like a simple challenge, which could be evaded by bad actors by obtaining names associated with phone numbers from leaked data. However, most of the robocallers currently make cheap mass robocalls. Hence, adding a simple challenge like the callee's name, adds cost to the malicious actors and works effectively to stop current robocalls. In the next chapter, we explore additional challenges, such as, interrupt and make conversation with the robocallers, ask further questions that are easy for a legitimate caller to answer but difficult for a robocaller. Such challenges would be difficult to break for a more sophisticated robocaller without AI capabilities.

The VA based defense proposed by us has a few limitations. It only works when the callee has a smartphone. The user study and performance evaluation experiments were conducted with a specific name set as the correct name. Since evaluating the correctness of keyword spotting algorithms is out of our scope, we did not conduct experiments with a

broader range of names. Also, our user study is limited to tech savvy university students. We hope the results of the study would be applicable to the broader population.

5.6 Conclusion

In this chapter, we proposed RobocallGuard, a smartphone virtual assistant (VA) that aims to automatically detect and block robocalls before they reach the targeted user. We developed an Android prototype app and conducted a user study, and showed that the VA can effectively block unwanted calls without disrupting the caller or callee experience. We also showed that our VA is able to correctly label 97.8% of robocalls without negatively impacting legitimate calls. We also discussed the limitations of RobocallGuard. In the next chapter we introduce SmartVA that can handle a more expanded threat model.

CHAPTER 6
VIRTUAL ASSISTANT MEDIATED INTERACTION FOR HANDLING TARGETED ROBOCALLS

6.1 Introduction

We introduced smartphone virtual assistant in the previous chapter and discussed how having a voice interaction model can detect unwanted calls without user interruption. However, the system we introduced is not effective against targeted robocalls and more sophisticated robocallers that might emerge in near future. In this chapter we expand our threat model to include targeted and more sophisticated robocallers and introduce a smarter Virtual Assistant equipped system that can engage in a more involved conversation with the caller. As in RobocallGuard, we want to preserve user experience and at the same time provide protection against mass, targeted and evasive robocallers.

As mentioned in the previous chapter, at a high level, the robocall problem resembles the email spam problem. However, a key difference in voice spam is that the content of the caller's message is not readily available. The call audio or context is only available after the call has been picked up and by the time the call is picked up the user is already exposed to the malicious actors. In response to this, recently, a number of automated caller engagement systems that attempt to analyze call content before forwarding a call have been proposed. However, such systems do not perform any blocking based on the call content. Therefore, they do not protect users from exposure to malicious actors. Robokiller [31], a smartphone application, on the other hand, features an Answer Bot that detects spam calls by forwarding all incoming calls to a server, which accepts each call and analyzes its audio to determine if the audio source is a recording. In an attempt to fool their victims current robocallers employ evasive techniques like mimicking human voice, not speaking

until spoken to etc. Hence, the defense mechanisms used by Robokiller are not enough to detect such evasive attackers.

In an attempt to protect users from mass, spoofed, targeted and evasive robocalls, we introduce Smart Virtual Assistant (SmartVA) named RobocallGuardPlus, a novel voice interaction model that can pick up incoming phone calls on behalf of the callee. It aims to make a natural conversation with the caller and filters out robocalls based on the conversation. While making conversation with the callers, RobocallGuardPlus asks questions that naturally occur in human conversations. The questions are designed in a way that it is easy and natural for humans to respond to; however, difficult for robocallers to provide an appropriate response without incurring significant additional cost. Based on the responses provided by the caller, RobocallGuardPlus uses a combination of NLP based machine learning to determine if the caller is a human or a robocaller. RobocallGuardPlus will only forward a call to the callee once it has determined that the caller is a human. Since RobocallGuardPlus does not solely depend on phone blocklists, it can stop caller ID spoofed mass robocalls. Moreover, there are incidents of targeted robocalls where robocallers ask for a particular callee. This is done to increase their credibility. The recent incidents of private data leak might increase such targeted robocalls. Since RobocallGuardPlus requires human-like interaction from the caller for their call to be passed, pre-recorded targeted robocalls can also be successfully blocked. To the best of our knowledge, we are the first to develop such a defense system that can interact with the caller and block robocalls even when robocallers utilize caller ID spoofing, target their victims and use voice activity detection to bypass the defense mechanism. All this is achieved without interrupting the callee and while preserving caller experience.

Although it may not be possible to stop all unwanted calls, we believe more secure communication via the telephony channel can be supported by an automated call screening agent that can detect and block such calls without significantly degrading user experience. In summary, we make the following contributions.

- We design a smart, interactive virtual assistant (RobocallGuardPlus) which can pick up and screen phone calls on behalf of the user. It tries to make a natural conversation with the caller and is designed to block prerecorded robocalls.
- RobocallGuardPlus uses a combination of NLP based machine learning models to determine if the caller is a human or a robocaller. To the best of our knowledge, we are the first to develop such a defense system that can interact with the caller and detect robocalls where robocallers utilize caller ID spoofing and voice activity detection to bypass the defense mechanism.
- To demonstrate the usability of our system, we have conducted an IRB approved user study to evaluate the user experience. The results from the user study demonstrate that the users had a positive experience and would benefit from using such a system.
- To conduct rigorous security evaluations, we recruited red team members to craft black-box attacks to defeat RobocallGuardPlus. Our red team members experimented with a sample from corpus of 8,000 real robocalls and showed that 95% of the mass robocalls, 82% evasive robocalls and 75% of targeted robocalls were not able to get to ring the phone, thus eliminating user interruption.

6.2 System Design

6.2.1 System Overview

Similar to RobocallGuard, RobocallGuardPlus receives incoming calls on behalf of the user without user interruption and intervention. If the incoming call is from a safelisted caller, it does not pick up the call and immediately notifies the user by ringing the phone. A safelist can be defined by the user, and can include the user's contact list and other allowed caller IDs (such as a global safelist which consists of public schools, hospitals etc.). On the other hand, if the call is from a blocklisted caller, RobocallGuardPlus blocks it and does not let the phone ring. However, if the caller ID belongs to neither a safelist nor a blocklist,

RobocallGuardPlus picks up the call without ringing the phone and initiates a conversation with the caller to decide whether this call should be brought to the attention of the user.

RobocallGuardPlus uses a combination of multiple techniques to detect robocallers. Upon picking up the call, it greets the caller and lets the caller know that he/she is talking to a virtual assistant. During the conversation, RobocallGuardPlus randomly chooses to ask a question from a predefined pool of questions that naturally occur in human conversations. It then determines if the response provided by the caller is appropriate for the question asked. It is difficult for a robocaller without natural language comprehending capabilities to provide an appropriate response but easy and natural for a human to answer these questions. RobocallGuardPlus determines whether the caller is a human or a robocaller based on the responses provided by the caller. The number of questions RobocallGuardPlus asks the caller before making this decision depends on the responses provided by the caller and the confidence RobocallGuardPlus has in labeling them as appropriate/not appropriate. For example, if RobocallGuardPlus is highly confident that the caller is a human or a robocaller after asking two questions, it chooses not to ask a third question. On the contrary, RobocallGuardPlus asks the next question if it is not able to make a decision at any current given time. To strike a balance between usability and security, the maximum number of questions RobocallGuardPlus asks before making a decision is five. The algorithm is explained in detail in the later sections. Based on the techniques discussed above, RobocallGuardPlus labels a caller as human or robocaller. If the caller is deemed to be a human, the call is passed to the callee along with the transcript of the purpose of the call. On the other hand, if the caller is determined to be a robocaller, RobocallGuardPlus blocks the call and notifies the user of the blocked call through a notification. RobocallGuardPlus also stores the transcript of the call for the user's convenience. Since our proposed solution does not depend on the availability of a blacklist of known robocallers, it can be effective even in the presence of caller ID spoofing.

6.2.2 Conversation Agent Challenges

RobocallGuardPlus can be thought of as a conversational agent that makes a quick conversation with the callers and makes a decision based on their responses. There has been a considerable amount of research conducted on conversational agents in the field of natural language processing [114, 115, 116]. Over the past few years, conversational assistants, such as Apple's Siri, Microsoft's Cortana, Amazon's Echo, Google's Now, and a growing number of new services have become a part of people's lives. However, due to the lack of fully automated methods for handling the complexity of natural language and user intent, these services are largely limited to answering a small set of common queries involving topics like weather forecasts, driving directions, finding restaurants, and similar requests. Conversational agents such as Apple's Siri demonstrated their capability of understanding speech queries and helping with users' requests. However, all of these intelligent agents are limited in their ability to understand their users and they fall short of the reflexive and adaptive interactivity that occurs in most human-human conversation [117]. Huang et. al. [118] discusses the challenges (such as identifying user intent, having clear interaction boundaries) associated with such agents.

RobocallGuardPlus consists of multiple modules that examine the caller's responses. These modules determine if a response is in fact an appropriate response to the question asked. Building natural language models presents numerous challenges. First, a large annotated dataset is required to build highly accurate NLP models. However, dataset consisting of robocall messages is limited and small in size. Moreover, human responses to secretary-like questions is also limited. Hence, building an effective virtual assistant from limited dataset becomes challenging. Second, models that have the capability of fully understanding natural language and user intent tend to be very complex and is still an area of ongoing research in the field of natural language processing. Also, we intend RobocallGuardPlus to be used real-time in a phone. Therefore the models should be lightweight which adds another challenge for us. Finally, most of the work on conversational agents has focused

on usability and how the conversation can be made more human-like. However, we need to strike the balance between usability and security since RobocallGuardPlus is designed to face both human callers and robocallers. Having the conversational agent succeed in an adversarial environment while at the same time being user-friendly to human callers is even more challenging.

6.2.3 Threat Model

In-scope Threats

In this section we describe the scope of threats that our RobocallGuardPlus is designed to protect against. Since it extends the functionality of RobocallGuard, all threats addressed by it are covered and we discuss the additional threats only.

Targeted Robocalls: Although not frequent, robocalls may be targeted. Targeted robocalls are when the robocallers know that name and phone number association of a particular victim. Hence they ask to speak with the callee by saying his/her name once their call is picked up. There have been multiple incidents of private leaked data which made it easier for bad actors to craft such targeted robocalls. RobocallGuardPlus can stop such unwanted targeted robocalls. Since RobocallGuardPlus demands human-like interaction from the callers to answer multiple questions, robocallers can not fool the defense system just by providing the callee's name.

Evasive Robocalls: We assume an evasive robocaller does not have AI capabilities, hence cannot comprehend what RobocallGuardPlus is saying; however has knowledge of RobocallGuardPlus's actions and tries to bypass it. The attack discussed here is currently not common, but may emerge in the future in an attempt to defeat intelligent phone call defense tools such as the one we propose in this paper. An evasive robocaller might utilize voice activity detection to identify interruption from the callee's side and pause accordingly to give the impression of liveness. It may also learn common questions and try to provide prerecorded responses in a certain order in an attempt to fool RobocallGuardPlus. How-

ever, without comprehending what the virtual assistant is saying, it is very difficult for the robocaller to provide an appropriate response to all questions and interact reasonably with RobocallGuardPlus. Since, we randomize the order of RobocallGuardPlus's questions and use a combination of multiple challenges to detect a robocaller, the possibility of providing a reasonable response and overcoming all challenges is very low for even an evasive robocaller. We discuss it in detail in the results section.

Out-of-scope Threats

RobocallGuardPlus does not protect against the following types of attacks.

Unwanted calls from AI equipped attacker: Attacks where the attacker is equipped with AI are not common in the phone fraud ecosystem. However, with the availability of tools like Google Duplex, attackers may be able to craft AI equipped attacks where robocallers make a natural conversation with the other party, and fool RobocallGuardPlus by pretending to be a human caller. Attacks where robocallers start interacting like humans and go to the extent where even humans have difficulty identifying between a robocaller and a human caller are out of scope. Moreover, building such an AI equipped attack is expensive and requires a substantial amount of resources. Since robocallers aim at making cheap mass calls so that they can reach a vast number of targets, crafting such AI equipped attacks will add significant cost for them.

Unwanted live calls from humans: Unwanted calls can be made by telemarketers, debt collectors or humans working for a scam campaign. Since RobocallGuardPlus looks for human-like interaction from the callers, such unwanted live calls from humans cannot be stopped. Since malicious actors who craft these campaigns aim at decreasing their cost and increasing the benefit incurred by fooling victims, having a human caller instead of a robocaller increases their cost significantly.

The threat model described above demonstrates the key differences between RobocallGuard and RobocallGuardPlus. Since robocall detection in RobocallGuard solely relies

on whether the callers knows the name of the callee or not, it cannot protect users against targeted attacks. RobocallGuardPlus on the other hand can detect targeted attacks because it does not rely on solely rely on the callee name and asks multiple questions. However, the goal of RobocallGuardPlus is to distinguish between human callers and robocallers and it requires human-like interaction form the callers for their calls to be passed. Therefore, it cannot protect users against unwanted human callers and Duplex-like AI quipped callers. We address the cost of letting calls from unwanted human callers and AI equipped callers pass in the later sections.

6.2.4 RobocallGuardPlus Use Cases

There can be two example use cases of RobocallGuardPlus we propose. RobocallGuard-Plus can be embedded with the Phone app where all incoming calls are handled locally. With each incoming phone call, RobocallGuardPlus examines the caller ID, interacts with the caller without making the callee aware. Once the call is deemed to be from a human, RobocallGuardPlus makes the phone ring and lets the caller reach the callee. It also provides the context of the call on the phone screen while ringing the phone, so that the callee has some meaningful information about the phone call.

One other scenario is where RobocallGuardPlus is hosted at an external server such as a network carrier. In that case, all incoming calls go through RobocallGuardPlus hosted at the server and it performs the above mentioned activities including caller ID checking and interacting with the caller. Once the call is deemed to be from a human, RobocallGuardPlus then forwards the call and the call context to the end user(callee in this case).

6.2.5 RobocallGuardPlus Workflow

Since RobocallGuardPlus handles all incoming calls, the phone does not immediately ring and notify the callee of an incoming call. RobocallGuardPlus makes a preliminary decision based on the caller ID of the incoming call. There can be three broad scenarios: (i) the caller

ID belongs to a predefined safelist (ii) the caller ID belongs to a predefined blocklist, or (iii) the caller ID does not belong to these predefined lists and thus is labelled as an unknown caller. If the caller ID is safelisted, RobocallGuardPlus immediately passes the call to the callee. RobocallGuardPlus blocks calls from blocklisted caller IDs and does not ring the phone. Additional analysis is conducted for the calls from unknown callers to understand the nature of the call. The following steps are followed to handle calls from unknown callers.

1. RobocallGuardPlus picks up the call, greets the caller and lets the caller know that he/she is talking to a virtual assistant. RobocallGuardPlus then randomly chooses to ask the caller to hold or continue the conversation.

2. Once the caller has responded to the previous question, RobocallGuardPlus then asks another randomly chosen question from the question pool described below. The question is chosen by RobocallGuardPlus according to rules defined later in this section. The questions are designed in a way which are easy and natural for humans to answer, but without comprehending what the question is, it is difficult for robocallers to answer. RobocallGuardPlus then determines if the response from the caller is appropriate or reasonable for the question asked and assigns a label (appropriate, not appropriate). RobocallGuardPlus also assigns a confidence score with each label.

3. RobocallGuardPlus then might ask another question or make a decision on whether the caller is a human or robocaller. The number of questions RobocallGuardPlus asks the caller before making this decision depends on the responses provided by the caller earlier and the confidence RobocallGuardPlus has in labeling each of them as appropriate/not appropriate. For example, if RobocallGuardPlus is highly confident that the caller is a human or a robocaller after asking two questions, it chooses not to ask a third question. On the contrary, RobocallGuardPlus asks the next question if it is not able to make a decision at any current given time. The minimum and maxi-

mum number of questions RobocallGuardPlus asks before labeling a caller human or robocaller is two and five respectively. It should be very natural for human callers to respond to these questions during a phone call. However, it is difficult for robocallers to act human-like in this scenario. Without comprehending what RobocallGuardPlus has asked them to do, it is not likely that they will be able to provide an appropriate response.

4. RobocallGuardPlus asks for the purpose of the call before completing the conversation with the caller if it has not already been asked. This question allows RobocallGuardPlus to provide additional context to the callee about the nature of the incoming call. It is important to note that all the questions discussed till now are asked in a random order. Therefore, an adversary cannot simply play pre-recorded responses in a certain order to fool RobocallGuardPlus.

5. Based on the steps discussed above and following the algorithm discussed in subsubsection 6.2.7, RobocallGuardPlus determines whether the caller is a human or not. If the caller is deemed to be a human, the call is forwarded to the user along with the content and other useful information (such as the name of the caller) about the incoming call. Calls from robocallers are blocked. Moreover, RobocallGuardPlus provides notification and information about the blocked call to the user.

6.2.6 RobocallGuardPlus Questions

During the conversation with the caller, RobocallGuardPlus picks questions to ask from Table 6.1. These questions are asked to determine if the caller can provide relevant answers to natural questions occurring in a typical phone conversation between two humans. The responses to these questions determine if the caller is a robocaller or a human. The questions are designed in a way that are easy and natural for a human caller to respond to during a phone call. However, the responses to these questions are specific enough that they do not

typically appear in robocall messages. While designing the questions, we aimed to balance between usability and security. The trade-off we intend to make depends on the following aspects.

- *Number of questions:* The virtual assistant based defense should strike a good balance between usability and security when determining the number of questions to ask the caller before making a decision about the call. Asking too many questions might annoy the caller and degrade the call experience significantly. On the other hand, asking too few questions can make it easy for the attackers to evade the system. Hence the virtual assistant should ask a number of questions that can ensure security and at the same time preserve user experience.

- *Type of questions:* Caller experience greatly depends on the type of questions that are asked. To preserve usability, we are limited to questions that are reasonably common at the beginning of a phone conversation between people who may not know each other. However, at the same time there should be enough variations in the questions so that the system is robust against attackers.

- *Gathering call context:* Besides preserving usability, determining the context of the call is another important goal. Therefore, context extracting questions should be asked so that meaningful information (such as the purpose of the call, the caller's name etc.) is available to the callee prior to engaging with the caller.

Following is the list of all questions RobocallGuardPlus asks.

Hold: As seen very commonly in phone conversations, RobocallGuardPlus asks the caller to hold briefly.

Context Extractor: This question is asked to extract context information such as purpose of the call. Pre-recorded robocalls will contain robocall messages (free cruise, vehicle warranty etc.); on the other hand human callers will provide legitimate context.

Name Recognizer Question: This question is asked to determine whether the caller can provide the correct name of the callee. If the caller knows the callee, they would be capable of saying who they are trying to reach.

Relevance Question: These questions are commonly occurring questions in a natural human conversation and allow RobocallGuardPlus to determine if the caller can actually comprehend the questions and provide appropriate responses.

Repetition Question: This question asks the caller to repeat what they just said. It is reasonable to expect from a human to repeat a statement by either repeating the exact same statement or saying a semantically similar statement. However, without understanding the question, robocallers won't be able to perform this task.

Speak up: It is very natural in a human conversation to ask a caller to speak up. RobocallGuardPlus asks this question to determine if the caller can indeed speak up when asked to do so.

Follow up: RobocallGuardPlus may choose to ask a follow up question after asking certain questions. For example, RobocallGuardPlus might ask "Can you please tell me more about it?" as a follow up question after asking "How can I help you?". Moreover after asking, "Who are you trying to reach?", RobocallGuardPlus might ask "Did you mean [name]?" as a follow up question. The [name] here can be two things, the correct name of the callee or an arbitrary name. For example if the name of the callee is *Taylor*, RobocallGuardPlus can ask "Did you mean Taylor?" or ask "Did you mean Tiffany?", where Tiffany is not the name of the callee.

It is important to note that RobocallGuardPlus uses multiple variations of each question. For example, the question "How are you?" can have multiple variations with the same meaning such as "How are you doing?", "How's it going?" etc. This enables us to defend against robocallers that can use the audio length of a question to determine what question was asked. With multiple variations of the same question, a robocaller truly needs to comprehend what RobocallGuardPlus is saying in order to provide an appropriate response.

Table 6.1: RobocallGuardPlus Question Examples

Hold	Please hold briefly.
Context Extractor	How can I help you?
Name Recognizer	Who are you trying to reach?
Relevance	How are you doing?
	How do you like the weather today?
Repetition	Sorry I couldn't hear you. Can you say that again?
Speak up	Can you speak up please?
Follow up	Can you tell me more about it?
	Did you mean [name]?

An AI equipped attacker might be able to automatically learn the questions. However, it is going to be difficult for the attackers considering the lack of significant amount of labeled data of such type of "secretary" conversations.

Question Order: In this section we discuss the rules RobocallGuardPlus uses to choose a question at each turn.

1. After the announcement and initial greeting by RobocallGuardPlus, it randomly chooses to ask the caller to hold or not.

2. RobocallGuardPlus then randomly chooses to ask the Context Extractor or Name Recognizer question with equal probablility.

3. At this point RobocallGuardPlus might choose to continue the conversation or block/forward the call based on the previous responses. If RobocallGuardPlus decides to continue the conversation at this point, it randomly chooses one of the Follow up, Relevance, Repetition, Name Recognizer, Hold questions with high probability or Speak up with low probability.

4. If RobocallGuardPlus decides to ask a fourth or fifth question, it randomly chooses one of the following questions with equal probability, Context Extractor, Repetition, Name Recognizer, Hold, Relevance, Speak up.

RobocallGuardPlus asks a specific question only once during the interaction with the caller.

The rules are designed to keep the conversation similar to a typical phone call in addition to increasing the entropy for the attacker so that there is no specific pattern that the attacker can exploit. An example conversation between a caller and RobocallGuardPlus is depicted in Figure 6.1.

SmartVA: Hello, you've reached the virtual assistant.
SmartVA: How can I help you?
Caller: I am looking for Taylor Smith. We met at a seminar today.
SmartVA: Before I forward your call, please
answer a few questions.
SmartVA: How are you doing today?
Caller: I'm fine.
SmartVA: I'm sorry I didn't get that. Can you
say that again?
Caller: I said I'm fine.
SmartVA: Before I forward your call, please
say your name.
Caller: This is Daniel.
SmartVA: Thank you. Please wait while I forward your
call.

Figure 6.1: SystemVA-Caller Conversation Example

6.2.7 System Architecture

In this section, we describe our system architecture (Figure 6.2) which is independent of the implementation environment. It is similar to RobocallGuard but includes a new Robocall Detector(RD) module. With each incoming call, the Metadata Detector module determines if the caller ID is present in the safelist or blocklist. Calls from safelisted callers are forwarded to the callee, calls from blocklisted callers are blocked and calls from unknown callers are passed to the Controller of the Robocall Detector.

Controller

The Controller has access to the question set and determines which question to ask the caller at every turn. After asking each question, the controller records the response from the caller. The audio from the caller is recorded until the caller finishes speaking or a maximum of 20 seconds, whichever is minimum. The audio recording is then transcribed

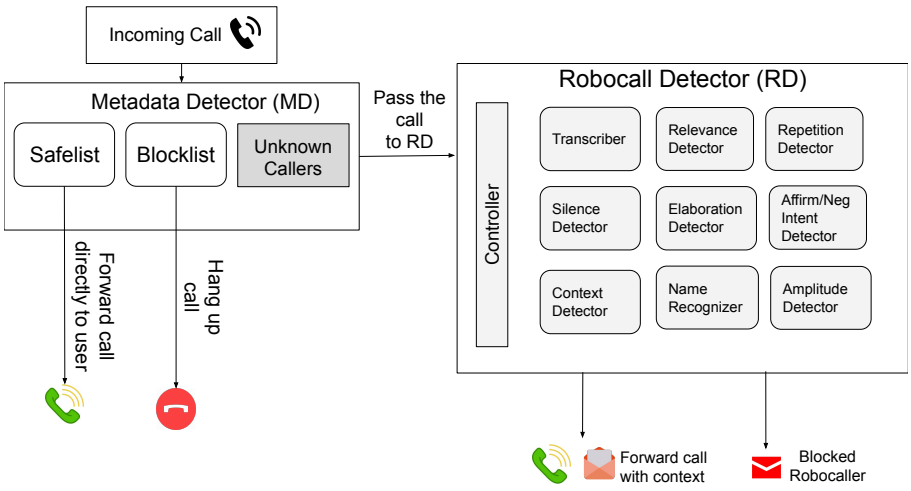


Figure 6.2: System Architecture

through the Transcriber module. The controller uses individual modules to label the transcripts (of responses) to determine if it is an appropriate response. For example Relevance Detector determines if the response is an appropriate one to the relevance question, Repetition Detector determines if the response is an appropriate one to the repetition question etc. Each of these modules (Relevance Detector, Repetition Detector etc.) predicts a label (appropriate/ not appropriate) and a confidence score with it. After every prediction, RobocallGuardPlus calculates a sequential probability ratio test (SPRT) score, S_i (where $1 \leq i \leq 5$) according to the following equation. This approach is inspired sequential probability ratio testing [119].

$$S_i = S_{i-1} + min((i/\lambda), 1) \times logA_i \tag{6.1}$$

$$logA_i = log(C_i/1 - C_i) \tag{6.2}$$

C_i is the confidence assigned by the corresponding module and λ is a tunable parameter that determines the weight of the i_{th} prediction. In our implementation we set λ to be 3. S_i determines the stopping rule of RobocallGuardPlus, i.e when to stop asking questions. As in classical hypothesis testing, SPRT starts with a pair of hypotheses, H_0 and H_1. We

specify H_0 and H_1 as follows.

$$H_0 : \text{Caller is human}$$

$$H_1 : \text{Caller is robocaller}$$

The stopping rule is a simple thresholding scheme:

- $a < S_i < b$: continue asking questions.
- $S_i \geq b$: Accept H_1
- $S_i \leq a$: Accept H_0

where a and b depend on the desired type I and type II errors, α and β. We choose α and β to be 5%.

$$a \approx \log \frac{\beta}{1-\alpha} \tag{6.3}$$

$$b \approx \log \frac{1-\beta}{\alpha} \tag{6.4}$$

RobocallGuardPlus requires at least two predictions and at most five predictions to make a decision on whether a caller is a robocaller or not. The controller implements the algorithm depicted in Figure 6.3 to make this decision. The algorithm also determines the number of questions RobocallGuardPlus asks before making this decision. At any given point, if a majority does not exist in the prediction labels, RobocallGuardPlus chooses to ask the next question. If a majority exists but the S_i score is between a and b, RobocallGuardPlus chooses to continue the conversation and asks the next question; otherwise RobocallGuardPlus checks if the majority labels and the label supported by SPRT (according to the stopping rule specified above) are in agreement. If yes, RobocallGuardPlus finalizes the label and makes a decision to forward the call if the caller is labeled human and block the call if the caller is labeled robocaller. If not, RobocallGuardPlus chooses to ask

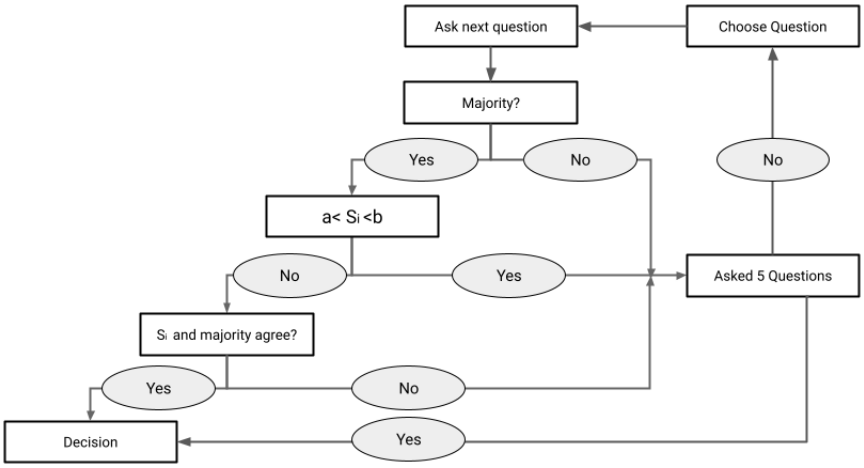


Figure 6.3: RobocallGuardPlus Algorithm

the next question. If RobocallGuardPlus is unable to make a decision after five predictions, it makes a decision of passing or blocking the call supported by the majority labels.

Transcriber

This component transcribes the responses provided by the caller. The transcriptions of the responses are then used by the other modules to determine if the responses are appropriate or not. Moreover, when RobocallGuardPlus notifies the user of an incoming call, the transcript of the conversation between the caller and RobocallGuardPlus is shown on the screen for additional context. This helps the callee to access the content of the call without picking up and engaging in the phone call. For calls which are not passed to the user and hung up by RobocallGuardPlus, the call context is saved for the user to review later. RobocallGuardPlus does not engage in a conversation with callers that are safelisted and passes the calls directly to the user. Therefore, transcripts are not provided for such calls. All other callers are greeted by RobocallGuardPlus and hence a transcript is made available to the user to understand the content of the calls.

There are many software libraries and APIs available for transcription. As explained

for RobocallGuard, we have chosen Google Cloud Speech API [105] because it has a very high accuracy and transcription can be performed in minimal time, unlike Kaldi [80] and Mozilla deep speech [106]. We send the audio recording of the caller's response to Google Cloud and a corresponding transcript is returned.

Silence Detector

This module is invoked by the controller when RobocallGuardPlus asks the caller to hold. RobocallGuardPlus randomly picks a hold time, t_s, ranging between five to ten seconds, asks the caller to hold and comes back to the caller to continue the conversation after t_s seconds. Human callers are expected to eventually stop talking when asked to hold and keep silent until the callee returns during a phone call. Therefore, this module detects if the caller has become silent during the t_s seconds RobocallGuardPlus asked them to hold. To determine whether the caller responded appropriately when put on hold, we determine if the the caller is silent during atleast half of the holding time, t_s. One approach to accomplish this is to employ Voice Activity Detection(VAD) to detect silence. However VAD can pick up any kind of audio including background noises and label it as voice. Since our goal to detect if the caller has stopped talking and kept silent when asked to hold, we take an alternative approach to detect that. The Silence Detector module transcribes everything said by the caller during the t_s seconds and calculates the average number of words said per second, w_{ps}. If w_{ps} is less than the threshold θ_s, the response is labeled as appropriate by the Silence Detector module and vice versa. We set θ_s in the following way. We calculate the average number of words spoken per second, aw_{ps} from our collection of pre-recorded robocall recordings and set θ_s as following,

$$\theta_s = (t_s \times aw_{ps})/2 \tag{6.5}$$

Context Detector

This module is invoked after the virtual assistant says "How can I help you?" Robocall messages are labeled as inappropriate response and everything else is labeled as an appropriate response to this question. To build such a classifier we use a dataset of phone call records collected at a large phone honeypot provided by a commercial robocall blocking company. This dataset contains 8081 calls (which had an average duration of 32.3 seconds) coming into a telephony honeypot during April 23, 2018 and May 6, 2018. It records the source phone number that made the call, the time of the call, the audio recording of the call and the transcript of the audio recording. Since it is a telephony honeypot, it contains some misdialed calls along with robocalls. To filter out misdialed calls, we use the approach described in chapter 3 to extract important topics from the transcripts of calls in our robocall dataset. Using LSI topic modeling, we extract 30 topics from our corpus of transcripts where each topic represents a spam campaign. We construct a similarity matrix by computing the cosine similarity between each transcript. We then convert the similarity matrix into a distance matrix by inverting the elements of the similarity matrix. We performed DBSCAN clustering on the distance matrix. DBSCAN is one of the most common clustering algorithms which given a set of points, groups together points that are closely packed together, marking as outliers points that lie alone in low-density regions. At the end of this, 72 clusters were created where each cluster represents groups of highly similar transcripts of robocalls. Since, the honeypot by nature contains unwanted calls, the clustering technique acts as a sieve that filters out all non-spam calls as outliers. We then take one representative robocall from each cluster and calculate the vector representations by projecting the robocall messages onto the pre-computed LSI topic model. To classify a response from a user, the Context Detector, after pre-processing the text, calculates the vector representation by projecting the response onto the pre-computed LSI topic model. It then computes the cosine similarity of the user response with pre-computed 79 roboball messages. If the cosine similarity is greater than a threshold, C_s it is labeled as an inappro-

priate response and vice-versa. In other words if the content of the caller response matches with any previously known robocall message, it is labeled as a not appropriate response; otherwise it is labeled as an appropriate response.

Elaboration Detector

A follow up question of "How can I help you?" is "Tell me more about it". This module determines if the response provided by the caller for this follow up question is appropriate or not. There has been a lot of work on text summarization in the area of natural language processing [120, 121, 122]. However a large number of data and complex architecture is required to train models accurate enough to be useful in real-life scenario. Therefore, to keep this component simple, we take the following approach. We count the numbers of words in the caller's response. If the number of words is higher than the number of words in the previous response, it is labeled as an appropriate response and vice versa. We understand that this is a naive approach and does not consider the semantic meaning of the responses, however, it is important to note that RobocallGuardPlus does not solely rely on the Elaboration Detector module, instead it uses the labels from multiple modules to make a final decision. Hence mislabeling by an individual component is compensated by the other components.

Relevance Detector

This module determines whether the response from the caller is an appropriate response for the Relevance Question asked by RobocallGuardPlus. We build a binary classifier which, given a (question, response) pair, labels the response *appropriate* if the response is a reasonable answer to the question selected by the controller and *not appropriate* if not. Human callers are expected to provide *appropriate* responses and robocallers are expected to provide *not appropriate* responses.

Dataset: To build such a classifier we use the "Fisher English Training Part 2, Tran-

scripts" dataset. Fisher English Training Part 2 Transcripts represents the second half of a collection of conversational telephone speech (CTS) that was created at the LDC during 2003. It consists of time-aligned transcripts for the speech contained in Fisher English Training Part 2, Speech. Under the Fisher protocol a large number of participants each make a few calls of short duration speaking to other participants, whom they typically do not know, about assigned topics. To encourage a broad range of vocabulary, Fisher participants are asked to speak about an assigned topic which is selected at random from a list, which changes every 24 hours and which is assigned to all subjects paired on that day. We further tailor this dataset to build our Relevance Detector model; we take the conversation between each speaker pair (speaker A and B) and convert it into (comment, response) pairs. Each of these (comment, response) pairs are labeled as appropriate. To generate the irrelevant examples, for each comment by speaker A we randomly pick a response which is not the response provided by the speaker B from the Fisher dataset and label that pair as not-appropriate. As a result, we generated 300,000 appropriate and 300,000 not-appropriate (comment, response) pairs to construct our dataset. We then perform sentence embedding on each data point to convert the text into a vector. Similar to word embeddings (like Word2Vec [123], GloVE [124], Elmo [125] or Fasttext [126]), sentence embeddings embed a full sentence into a vector space. We use Infersent [127] to perform sentence embedding on our data points. InferSent is a sentence embedding method that provides semantic sentence representations. It is trained on natural language inference data and generalizes well to many different tasks. Hence, the data points (comment, response) pairs are converted to (comment embedding, response embedding) pairs (where comment embedding denotes the sentence embedding of the comment and response embedding denotes the sentence embedding of the response). The (comment embedding, response embedding) pairs are then passed to the binary classification model we built.

Base Model: We used a Multilayer Perceptron (MLP) as our base model. We empirically set the architecture of our model as (1024, 512, 256, 1). 384,000 data points are used

to train the base model. The train, validation and test accuracy of the base model is 83%, 70% and 70% respectively. To test with robocalls we take the following approach. We treat the questions asked by the VA as a comment and the transcripts from robocall recordings as a response. However the base model performs poorly when tested with robocalls. Since the model is not specifically trained to recognize robocalls the testing accuracy decreases in this case.

Finetuning Base Model: We further finetune our base model to specifically recognize robocalls and legitimate (human) calls. Our goal is to build a model that can detect appropriate responses to the questions asked by the VA. We assume that human callers will be able to provide appropriate responses to the questions whereas robocallers will not. Therefore we label (question, robocall response) pairs as ”not appropriate” and (question, human response) pairs as ”appropriate” to finetune our base model.

Data collection and processing: To generate our ”not appropriate” responses, we use the dataset of robocalls described in subsubsection 6.2.7. We take the first 30 words (as we let each response to be of at most 20 seconds) from each robocall transcript and pair it with both relevance questions to form our ”not appropriate” responses. In this way we get 67 unique (question, robocall response) pairs. Since this dataset is too small to finetune a model and the number of unique robocall messages is limited, we perform data augmentation on the 67 unique robocall responses. For each robocall response we generate two more augmented text using the techniques in [128]. This yields 201 (question, response) ”not appropriate” pairs for each question from the Relevance question pool. To generate the appropriate pairs, for each question from the Relevance question pool we use quora to collect appropriate human responses to these questions. We augment the (question, human response) pairs in the same way. Upon generating the appropriate and not appropriate pairs we generate the sentence embedding pairs in the similar fashion described above. The (question embedding, response embedding) pairs are then passed to finetune our base model. Table 6.2 shows the test accuracy of the finetuned model.

Table 6.2: Relevance Detector Results

	First 10 words	First 20 words	First 30 words	First 40 words
Overall Accuracy	91.02%	97.18%	98.32%	97.21%
Robocall Accuracy	92.5%	98.46%	100%	98.49%
Human Call Accuracy	87.23%	93.62%	93.62%	93.62%

Repetition Detector

This module is invoked by the controller after the virtual assistant asks the caller to repeat what he/she just said. Once the caller has done responding to the repetition request by RobocallGuardPlus, Repetition Detector (RD) compares the caller's current response to the immediate last response to determine if the current response is a repetition of the immediate last response. To accomplish this task we build a binary classifier which, given a (current response, last response) pair, assigns the label "appropriate" if current response is a semantic repetition of last response and "not appropriate" if not.

Dataset: To build such a classifier, we collect (current response, last response) pairs from Lenny [129] recordings. Lenny is a bot (a computer program) which plays a set of pre-recorded voice messages to interact with spammers. Although not based on any sophisticated artificial intelligence, Lenny is surprisingly effective in keeping the conversation going for tens of minutes. There are more than 600 publicly available call recordings where Lenny interacts with human spammers (telemarketers, debt collectors etc.). During the conversation, Lenny asks the callers to repeat themselves multiple times. Among 600+ publicly available call recordings, we randomly select 160 call recordings and manually transcribe the parts where the callers have repeated themselves. Specifically we create 160 (current response, last response) pairs and assign them with the "appropriate" label. Since the telemarketers talking to Lenny are human callers, when asked to repeat themselves, they provide a semantic if not the exact repetition of their last statement. We expect most legitimate human callers to behave in the same way. Robocallers on the contrary are not expected to provide an appropriate response when asked to repeat what they just said. To

generate our "not appropriate" (current response, last response) pairs, for each last response we randomly pick a current response from the Lenny transcripts which is not an appropriate repetition. In this manner we generate 160 not appropriate pairs.

Repetition Classifier: We extract the following three features from the data points generated.

Cosine similarity: We calculate the cosine similarity between current response and last response.

Word overlap: Upon removing stop words and punctuation, we calculate the number of words overlapped between current response and last response.

Named entity overlap: Upon removing stop words and punctuation, we calculate the number of named entities in current response and last response. In information extraction, a named entity is a real-world object, such as persons, locations, organizations, products, etc., that can be denoted with a proper name. We use Spacy [130] to extract the named entities and then calculate the number of named entities overlapped between current response and last response.

These simple yet effective features allow us to determine if a $statement_1$ is a semantic repetition of $statement_2$ without using resource intensive machine learning models. We train 5 different classifiers using the above mentioned three features. Table 6.3 shows the test accuracies and false positive rates for each classifier. Table 6.3 also shows how the classifier performs on the robocall test set. To generate the robocall test set we take 79 representative robocalls messages and generate (current response, last response) pairs by setting the first sentence and second sentence from the the robocall messages as current response and last response respectively. Since, in this dataset none of the current responses are semantic repetitions of last responses, these pairs should be labeled as not appropriate. Since Random Forest has the highest robocall test accuracy and lowest false positive rate, it is chosen to be the most suitable classifier for the RD module.

Table 6.3: Repetition Detector Results

	Test Accuracy	Robocall Test Accuracy	False Positive Rate (FPR)
SVM	94%	82%	12.5%
Logistic Regression	93.6%	85%	12.5%
Random Forest	95%	86%	6.25%
XG Boost	95%	85%	9.4%
Neural Network	93.7%	83%	12.5%

Name Recognizer

We use the NR module, described in the previous chapter for RobocallGuard, which is a keyword spotting algorithm which can detect the right keyword. In our scenario, the correct name(s) of the callee is the keyword. We chose Snowboy to recognize the name. We treat Snowboy as a blackbox, which when provided with 3 audio samples, creates a model to detect the keyword. We embedded the downloaded trained model with the NR module to recognize the correct name(s). Since snowboy does not provide a confidence score we set the accuracy of snowboy (0.83) as its fixed confidence score for every label.

Affirmative/Negative Intent Recognizer

A follow up question of "Who are you trying to reach?" is asking the caller to confirm the name. RobocallGuardPlus does this in two ways, asking the caller to confirm by saying the correct name and saying an incorrect name. For example, if the correct name is Taylor, the virtual assistant will say, "Did you mean Taylor?" and expect an affirmative answer from a human caller. An alternative question is asking the caller to confirm the name by intentionally saying an incorrect name, such as, "Did you mean Tiffany?". In this case RobocallGuardPlus expects a negative answer from an human caller. Based on the question and the expected response from the caller, Affirmative/Negative Intent Recognizer labels a response from the caller as inappropriate and appropriate. To detect if the response is affirmative or negative we take the following approach. We manually compile a list of

affirmative (e.g. yes, yeah, true etc.) and negative (e.g. no, not etc.) answers. If an affirmative answer is expected and the caller's response contains any of the affirmative words, it is labeled as an appropriate response. Similarly, if a negative answer is expected and the caller's response contains any of the negative words, it is also labeled as an appropriate response. All other cases are labelled as inappropriate responses.

Amplitude Detector

The module is invoked when RobocallGuardPlus asks the caller to speak up. The Amplitute Detector determines if the caller has spoken louder and to do that it measures the average amplitude of the audio of the caller's response. If the average amplitude is higher by an error margin than the caller's previous response, Amplitute Detector labels it as an appropriate response and vice versa.

6.3 Evaluation

In this section, we report the results of the evaluations we conducted to measure the accuracy of decisions made by our RobocallGuardPlus. We also conduct a red team style security analysis and discuss the black box attacks the read teams crafted in an attempt to fool RobocallGuardPlus. By performing these experiments we demonstrate RobocallGuardPlus's effectiveness against robocalls in our threat model. We also discuss a user study that was conducted to evaluate the usability of our system.

6.3.1 Usability Study

RobocallGuardPlus is designed to provide the convenience of a human assistant while detecting robocalls. It also provides context for calls, which helps the callee decide if a call needs his/her attention. One of the most important goal behind our design choices of the system is to preserve user experience. To explore the usability of the system we introduce, we conducted an Institutional Review Board (IRB) approved user study. In the following,

we first describe the study setup, its participants and then discuss the results.

Study Setup

Our study participants consists of 20 users who were sampled from a population of college students and their families. All participants were required to be above 18 years old and fluent in English. We briefed the participants about the experiment process and explained the purpose of RobocallGuardPlus. Due to the ongoing pandemic, it is not safe to conduct an in-person user study. To ensure that the user study avoids physical contact, we hosted RobocallGuardPlus on a AWS server which can be accessed via a web interface. Upon recruiting the users, we provide an URL which directs them to a web interface in their browser. By clicking on a Start Call button, users initiate an interaction with RobocallGuardPlus. This action mimics starting a phone call and reaching the virtual assistant. The users are able to talk to RobocallGuardPlus through the microphone of their own device. In this study all users played the role of a caller and made four calls on various given topics. Such a call took at most one minute. Upon completing each emulated phone call, the users are provided with a set of survey questions that focus on evaluating the user experience. At the end of the user study, each user was asked three generic questions about their overall experience with RobocallGuardPlus.

User Actions

In this section, we describe each task the participants performed during the user study in detail. We performed two experiments, one where the callers know the name of the callee and one where the caller doesn't know the name of the callee. During the first experiment, we preset the correct name to be *Taylor* instead of having each user set a name. We make this choice because the purpose of the user study is to get insights about call experience in the presence of a virtual assistant, rather than testing the accuracy of the keyword spotting algorithm. We recruited 15 out of our 20 users for this experiment and provided the follow-

ing four topics to make the four simulated phone calls. The topics are selected such that it is natural for a phone call setting and common in real life scenarios. We choose the last two topics to be in overlap with robocall topics (free cruise and car warranty). Since human callers are interacting with RobocallGuardPlus here, it is expected that the calls should be forwarded even when the call topics are overlapped with the robocall topics. This provides evidence that our system does not conservatively block calls containing words that might be present in robocall messages.

1. Make a call to your friend Taylor to make movie plans.
2. Make a call to your doctor Taylor to make an appointment.
3. Make a call to your friend Taylor to plan a cruise vacation plan.
4. Make a call to your mechanic Taylor about your car warranty.

During the second experiment, the caller is either given an incorrect name or no name at all. The participants had no idea about what the correct name was. We recruited the remaining 5 users for this experiment. Since in a real life scenario most legitimate human callers know the callee's name, we have a lower number of users playing the role of a caller who does not know the callee. Following is the call topics of the three calls made by the callers during the experiment. The caller is not given any name in the first two topics and is given an incorrect name in the last two topics. Since RobocallGuardPlus requires human-like interaction to forward a call, it is expected that the calls will be forwarded even if the caller doesn't know the correct name. This ensures that calls from first-time legitimate callers who don't know the name of the callee are not blocked.

1. Play the role of a telemarketer and make a call to sell a computer.
2. Make a call to conduct a survey on robocalls.
3. Make a call to your friend Robert to meet for lunch.
4. Make a call to Jordan about your car warranty.

User Study Demographics

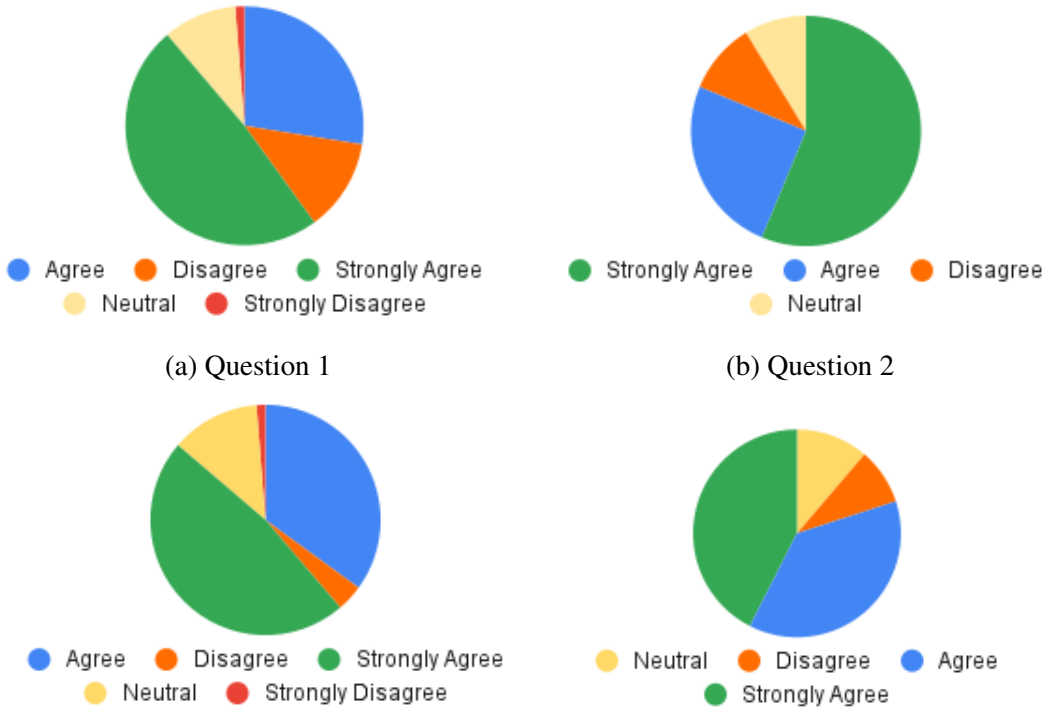
Most of our users were aged between 20 o 35 years old. 40% of our users were female and the rest were male. We collected information about phone usage and previous experience regarding robocalls from our users. 60% of our users user Android and the rest use iPhone. Moreover, 40% of our users reported that they use some sort of call blocking applications (e.g. Truecaller, Youmail etc.). 82% of the users who use call blocking applications reported that they never pick up calls labeled as suspicious/spam. Regarding their previous experience of receiving robocalls, 35% of our users reported that they receive one or more robocalls every day and 50% of the users receive one or more robocalls every week. However the majority of these users don't use call blocking applications and are unprotected from robocallers.

User Study Survey Results

In this section we present the results obtained from the user responses to the survey questions. After each call the users were asked the following four questions. The user responses to these questions are summarized in Figure 6.4

1. Question 1: The conversation with RobocallGuardPlus felt natural.
2. Question 2: I was able to answer the questions asked by RobocallGuardPlus without difficulty.
3. Question 3: The number of questions I had to answer was acceptable.
4. Question 4: The time I spent interacting with RobocallGuardPlus before my call was forwarded/blocked is acceptable.

Figure 6.4(a) demonstrates that most of the users reported that the conversation with RobocallGuardPlus felt natural. Only 14.7% users reported that the conversation did not



(a) Question 1

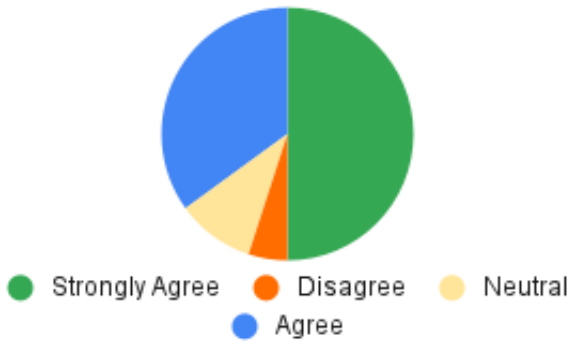
(b) Question 2

(c) Question 3

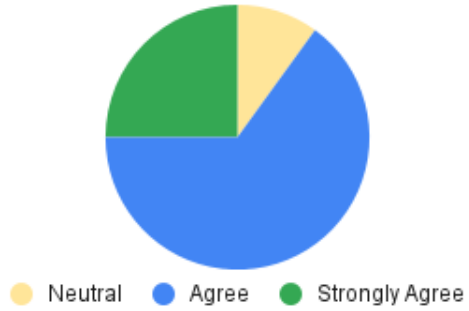
(d) Question 4

Figure 6.4: User Responses

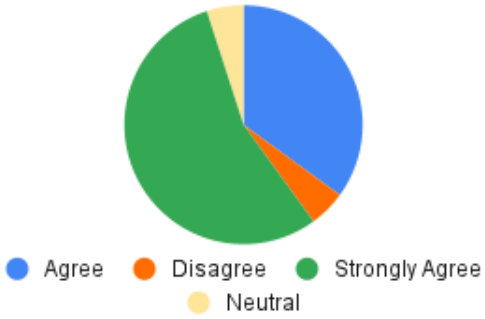
feel natural. We collected additional feedback about their experience and asked for suggestions from the users during the study. The users that felt that the conversation was not natural, mentioned that when they responded to how they were or how the weather was, they also asked the virtual assistant the same question. For example, they responded, "I am fine. How are you?". However, RobocallGuardPlus did not respond to their question and moved on to ask the next question. It is understandable because RobocallGuardPlus is not designed to respond to the caller's questions. This feedback from the users was useful and could be incorporated in future work. Figure 6.4(b) demonstrates that most of the users (81.3%) were able to answer the questions asked by RobocallGuardPlus without difficulty. Only 10% users reported that they had difficulty answering the questions. The additional feedback collected from our users showed that 2 out of 20 users mentioned that they felt unfamiliar with the system and had difficulty answering the questions during the first call. However after making one or two calls they felt familiar and were able to answer the ques-



(a) Question 5



(b) Question 6



(c) Question 7

Figure 6.5: User Responses (contd.)

tions with ease. We also asked users if the number of questions they had to answer was acceptable. Only 5% users reported that it was not acceptable (Figure 6.4(c)). We found out that these users had to answer five questions before a decision about their call was made. Moreover, we computed the number of questions RobocallGuardPlus asked during it's interaction with our users and found out that in 67% of the cases RobocallGuardPlus made a decision by asking upto three questions. Hence, 83% of the users reported that the number of questions they had to answer was acceptable. Figure 6.4(d) further shows that only 8.8% users felt that the time they spent interacting with the virtual assistant before their call was forwarded/blocked is not acceptable.

After the end of the experiment each user were asked the following three questions regarding their overall experience with RobocallGuardPlus. The user responses to these questions are summarized in Figure 6.5.

1. Question 5: It was easy to interact with the RobocallGuardPlus.

2. Question 6: I felt comfortable with RobocallGuardPlus intervening phone calls.

3. Question 7: I think I would like to use a system equipped with such a Virtual Assistant frequently.

It is depicted in Figure 6.5(a) that 85% of the users reported that it was easy to interact with RobocallGuardPlus. As mentioned earlier, the interaction with RobocallGuardPlus becomes easier as the users get familiar with the system after a couple of calls. Moreover, 90% of the users reported that they felt comfortable with RobocallGuardPlus intervening the calls (Figure 6.5(b)) and 90% of the users reported that they would like to use a system equipped with a Virtual Assistant frequently (Figure 6.5(c)).

6.3.2 Measuring False Positives

We define false positives as the percentage of calls from human callers that were mistakenly blocked. Since the goal of RobocallGuardPlus is to block robocallers and forward calls from human calls, false positives represent the calls from human callers that were deemed as robocallers and thus were mistakenly blocked. To compute our false positive, we use the data collected during our user study. 20 users made 80 calls in total and only 7 calls were blocked, yielding an overall false positive rate of 8.75% . We further investigated the details of the calls that were mistakenly blocked and found that in 5 out of the 7 (71%) blocked calls the user kept silent and didn't answer all the questions, hence the calls were blocked. This is expected because RobocallGuardPlus blocks calls if human-like interaction is not detected. Therefore, the calls where the users didn't respond to the questions asked by RobocallGuardPlus were blocked. We also investigated if not knowing the right name incurs a higher false positive rate. 5 out of the 20 users were not provided with the correct name "Taylor". Among the 20 calls made by users who did not know the correct name, 2 calls were blocked. Hence the false positive rate (10%) did not drastically increase for

users who did not know the correct name. This demonstrates that the system is not heavily dependent on the caller knowing the name of the callee. This ensures that legitimate callers who might not know the correct name can also reach the callee. We also investigated if the call topics had any effect on which calls should be forwarded. Two of our call topics in the user study overlapped with robocall topics (free cruise and vehicle warranty). 35 calls were made by our users regarding these call topics and only 3 calls were blocked, yielding a false positive rate of 8.6%. It is important to note that these 3 calls are the same calls where the users did not respond to all questions. Therefore, it can be said that RobocallGuardPlus is not biased towards the specific keywords (e.g. cruise, vacation, vehicle, warranty etc.) and does not blindly block calls when it detects such keywords. Instead it looks for human-like interaction and blocks calls when it cannot detect such responses. Our results demonstrate that most of the false positives occur when the caller does not respond to the questions asked by RobocallGuardPlus. Naturally, a low false positive rate is required for defense against robocallers to be useful in practice. It can be expected that as users become more familiar with the system and respond to all questions, the false positive rate would further decrease. It is important to note that calls from whitelisted known callers will not be intervened by RobocallGuardPlus hence their calls will never be blocked by the virtual assistant. Thus the false positives will only include calls from unknown callers. In addition, blocked callers are asked to leave a voicemail, which is similar to the callee not picking up the call. Since this is common for unknown caller IDs, the cost of an false positive is expected to be very low. While not comprehensive, we believe that results reported from the user study data represents a meaningful and reasonably approximate estimate of the false positive rate.

6.3.3 Security Analysis

In this section we discuss the security evaluations we conducted to measure the robustness and correctness of RobocallGuardPlus. We first report the effectiveness of RobocallGuard-Plus against current robocalls and a baseline attack in which the robocaller randomly re-

sponds to its questions. For additional security analysis, we recruited a group of graduate students with varying expertise in security who play the role of an attacker. We had two red teams, Red Team A and Red Team B working independently of each other. Red Team A consisted of two masters student and one one PhD student. Red Team B consisted of one masters student with an expertise in voice based attacks. Both our red teams crafted black-box attacks. They were provided with unlimited access to RobocallGuardPlus, however were not provided with the details of the system. Our red teams were encouraged to extract as much information as possible from interactions with RobocallGuardPlus, however how the system works was not shared with them. All attacks made by our red teams was automated. In other words no human interaction was present when making conversation with RobocallGuardPlus. Our red teams mainly crafted three types of attacks discussed below.

Current Mass Robocall Attacks

We define mass robocalls as automated calls made by attackers who don't have any specific information, such as name, about their target victim. We provided Red Team A with 72 representative robocall recordings from a corpus of 8000 real robocalls. The sample was selected using the method described in subsubsection 6.2.7. Red Team A used the robocall recordings in various scenarios to craft the black-box attacks on RobocallGuardPlus. We report the findings of Red Team A in this subsection.

In the first experiment, Red Team A used the robocall recordings and played them as soon as RobocallGuardPlus picks up the call. Each recording was played two times to make two independent calls. Red Team A found that 95% of the 144 mass robocalls were successfully blocked. In the second experiment, Red Team A used the same robocall recordings. However, they did not play the recordings as soon the call is picked up. Instead the recordings were played after RobocallGuardPlus finishes saying the greetings and asks the first question. This was done to simulate the evasion technique many current robocallers use, where they speak only after being spoken to, once the call is picked up. Red Team A

similarly used every recording two times to make two independent calls and found that 94.5% of the 144 mass robocalls were successfully blocked. Red Team A then conducted additional experiments by increasing and decreasing the playback speed of the audios and did not find any difference in RobocallGuard's performance. Moreover, they played the shorter robocall recordings in a loop and found similar success rate. Red Team A reported that they could not find any pattern for current mass robocalls which can be exploited to attack RobocallGuardPlus.

Baseline Random Response Evasive Robocalls Attack

We define our baseline as an adversary who randomly guesses a response to the questions asked by RobocallGuardPlus. We assume that the random attacker has extracted information about the the questions asked by interacting multiple times with the virtual assistant. Furthermore, we assume that the random attacker has pre-curated responses to all the questions of the question pool and randomly selects a response to play when conversing with RobocallGuardPlus. To build our random adversary we take the following approach. We create appropriate responses for the questions from the question pool. For example, we create the response "I am fine." for the question "How are you doing?". We assume that the random adversary does not craft targeted attacks, hence, does not know the correct name. Therefore, we create the response, "I am trying to reach Mike." as an answer to the question "Who are you trying to reach?", where Mike is a random common name in the US. As an appropriate response for the Hold question the random adversary randomly chooses to pause between 5 to 10 seconds. Also, we don't create any response for the *Repetition* and *Speak up* questions as these responses are related to the previous response. Once the response pool is created, the adversary makes a call to the RobocallGuardPlus and then randomly chooses the number of questions to answer. We assume that the random adversary has extracted the information that RobocallGuardPlus asks 2 to 5 questions. Therefore, the random adversary can choose between 2 to 5 responses when talking to the virtual assistant.

During the conversation, the adversary randomly chooses a response from the pre-curated response pool. In this way, the random adversary created by us made 135 calls to RobocallGuardPlus where the followings choices were made randomly by the adversary during each call: (i) The number of questions to answer, (ii) The response to a certain question, and (iii) the time interval between each response. 123 out of 135 calls were blocked by RobocallGuardPlus yielding a blocking rate of 91.1% for random attackers. We define this as our baseline and evaluate RobocalGuardPlus's effectiveness against smarter and more evasive attackers in the following section.

Red Team Created Evasive Robocall Attacks

We define evasive robocallers as attackers who use information extracted from interacting with RobocallGuardPlus to craft black-box attacks. Red Team B was involved in making the evasive robocall attacks. To craft the evasive attacks Red Team B interacted with RobocallGuardPlus several times to extract information about which questions are asked, the order and frequency of the questions. Then they created audios of appropriate responses to the questions. For the Name Recognizer question, they used the callee names "Jessica" and "William". For the questions "What's the weather like?" and "How are you?", they used "Great" as a general answer. For the Context Detector question, the sentence pattern they used is "I want to talk to CALLEE NAME". Red Team B realized that once the call is picked up RobocallGuardPlus will either ask the caller to hold briefly or ask questions directly. They created several audios where some start with a pause and others start with the responses to the questions. Red Team B sorted the answers in different orders and created 8 different audios. They used each audio 10 times to make 80 calls in total and found that RobocallGuardPlus was successfull in blocking 82% of the evasive attacks. The only times the attacks were successful was when the order of the responses aligned perfectly with the questions which happened 18% of the times. Since the attacks were only successful by random chance, it can be said that RobocallGuardPlus is effective against evasive attacks.

Targeted Robocall Attacks

We define targeted robocallers as attackers who target their victims individually and use some specific information about their victim to their benefit. In order to conduct targeted attacks we shared the correct name of the callee, "Taylor", with our red teams so that they can use this information in their attacks.

Red Team A conducted a limited scale targeted attack on RobocallGuardPlus. First, they took the three most successful mass robocall recordings (audios of calls that were not blocked in the previously discussed mass robocall attacks), transcribed them, and re-recorded them by adding the word "Taylor" in appropriate sentences to implement a targeted attack. They ran the 3 recording 27 times and found out that adding the name did not provide any extra leverage to their attacks. Finally, they took the 3 most unsuccessful mass robocall recordings (audios of calls that were always blocked in the previously discussed mass robocall attacks), transcribed them, and re-recorded them by adding the word "Taylor" in appropriate sentences. They ran the 3 recording 27 times and found out that RobocallGuardPlus blocked 75% of the targeted calls.

Successful Attacks

Both our red teams found a weakness in the RobocallGuardPlus which can exploited to craft a successful attack. Our red teams reported that if the a short generic response such as "I want to talk to Jessica" is played for every question, RobocallGuardPlus fails to block the call majority of the times. This occurs because such a short generic response is applicable for many of the questions RobocallGuardPlus asks in the beginning of the call, such as "How can I help you?", "Can you please hold?", "Can you repeat?". This attack can be countered by adding a simple check to see if the caller is saying the same thing over and over again.

6.4 Discussion and Limitations

RobocallGuardPlus is designed to protect users against current mass robocallers and more sophisticated robocallers that might emerge in future. Our user study shows user experience is preserved for callers when they interact with the virtual assistant. However, there are a few limitations. RobocallGuardPlus can not block calls from AI equipped robocaller who can comprehend what the virtual assistant is saying and respond accordingly. Google Duplex is an example of such AI equipped automated caller. However it should be kept in mind that developing and maintaining such AI enabled attacked can be expensive and resource-intensive. RobocallGuardPlus is also not designed to protect against unwanted human callers such as telemarketers, debt collectors etc. Therefore spam campaigns that hire human callers to conduct their attacks can not be stopped by RobocallGuardPlus. Nevertheless hiring human callers is also expensive for malicious actors.

The user study and security analysis experiments were conducted with a specific name set as the correct name. Since evaluating the correctness of keyword spotting algorithms is out of our scope, we did not conduct experiments with a broader range of names. Also, our user study is limited to college students and their families. We hope the results of the study would be applicable to the broader population.

Since RobocallGuardPlus is not designed to answer questions from callers, the natural flow of the conversation may be obstructed as pointed out by some of our users. In our future work we plan to explore if RobocallGuardPlus can ask dynamic questions instead of static questions chosen from a question pool. In that case, similar to a chatbot [131], RobocallGuardPlus can carry a more natural conversation with the caller. The responses by RobocallGuardPlus can be generated using natural language generation. Such a method would further improve the robustness since what RobocallGuardPlus says would depend on the caller's response. Moreover, the flow of the conversation would be more natural which would further enhance the user experience.

6.5 Conclusion

In this chapter, we expand our threat model and introduce RobocallGuardPlus, a smart virtual assistant(SmartVA) system that aims to automatically detect and block sophisticated robocalls before they reach the user. We discuss the design of our RobocallGuardPlus and explore if it can be effective against mass, targeted and evasive robocallers. We developed a proof-of-concept system, hosted it on a AWS server and conducted a user study to access the usability of such a system. The results from the user study demonstrate that RobocallGuardPlus preserves user experience and at the same time keeps the false positive rate low. To conduct security evaluations, we recruited multiple red teams with various expertise who crafted back-box attacks on RobocallGuardPlus. Our red teams reported that 95% of the mass robocalls were successfully blocked. The red teams also developed sophisticated robocallers and found out that RobocallGuardPlus was successful against 82% evasive attacks.

CHAPTER 7
CONCLUSIONS AND FUTURE WORK

7.1 Dissertation Summary and Contributions

Telephone scams are now on the rise and without effective countermeasures there is no stopping. Unwanted robocalls have become such a serious problem that people often no longer pick up the phone when it rings. While many of us may not believe the free cruise offers or the IRS lawsuit threats, senior citizens and new immigrants often fall victim to such scams. Some of the users might be aware enough not to be fooled by the scams but they cannot stop their phone from ringing by annoying robocalls. Our research is aimed at bringing trust back to the telephony channel and making a better telephony experience for everyone. The contributions made by this work include:

- We characterize the threat landscape that currently perpetrates telephony abuse.
- We present the first systematic study estimating the effectiveness of phone blacklists. We analyze the characteristics of multiple data sources that may be leveraged to automatically learn phone blacklists, and then measure their ability to block future unwanted phone calls.
- We evaluate the effectiveness of the phone blacklists we were able to learn, and show that they are capable of blocking a significant fraction (e.g., more than 55% of unsolicited calls) of future unwanted calls in the current situation. However, their performance decreases as caller ID spoofing increases.
- We explore an automated voice-based interaction approach that maintains both caller and callee user experience, eliminates user interruption and stops unwanted calls even in the presence of spoofed calls. Although it may not be possible to stop all

unwanted calls, we believe more trusted communication via the telephony channel can be supported by an automated call screening agent that can detect and block such calls without degrading user experience.

- We evaluate a call screening virtual assistant that uses automated call handling to defend against robocalls and other types of spam calls, including those that evade blacklists with caller ID spoofing. In addition, transcription of call audio recorded by the virtual assistant is used to provide meaningful context about incoming calls to a user when the phone rings.

- To demonstrate the ability of the virtual assistant to detect robocalls, we have developed a proof-of-concept smartphone app named RobocallGuard. To this end, we experimented with a corpus of 8,000 real robocalls collected by a large phone honeypot, and show that all of them can be detected and thus blocked.

- In addition, our proof-of-concept app allowed us to conduct an institutional review board (IRB) approved user study to assess the usability of our virtual assistant. The results of this study demonstrate that the experience of a typical phone call is preserved for both callers and receivers, while benefiting from the ability to detect robocalls and other potentially unwanted calls.

- We improve the robustness of our virtual assistant by expanding our threat model. We introduce a voice interaction enabled smart agent (RobocallGuardPlus) which can initiate a conversation with the caller and can successfully identify sophisticated robocallers. RobocallGuardPlus makes a natural conversation with the caller and require human-like interaction from the caller for their call to be forwarded. RobocallGuardPlus asks questions that occur naturally in human conversations.

- We develop a combination of NLP based models which can determine if the caller is a human or robocaller.

- We develop a smart virtual assistant that can interact with the caller and detect even targeted robocalls where robocallers utilize caller ID spoofing and voice activity detection to bypass the defense mechanism.

- We conduct an IRB approved user study to asses the usability of RobocallGuardPlus. The results from our user study show that the users had a positive experience while interacting with RobocallGuardPlus and 90% users reported that they would like to use such a system frequently.

7.2 Discussion and Limitations

In this dissertation we focused exclusively on telephony abuse due to robocalls that deliver voice spam. Although robocalls are the most pervasive forms of abuse through telephony channel, spamming through SMS is also common. We exclude such attacks from our research. The defenses proposed by us mostly rely on the availability of a smartphone. We envision end users using smartphone apps such as call blocking apps or RobocallGuard to prevent robocalls from reaching them. To protect users using landline phones from unwanted calls, our defense solutions can be hosted on network carriers instead on end user's devices. However, this may require regulatory changes that allow carriers to prevent calls deemed as unwanted from reaching the end user.

We explore a broad threat landscape and we expect our defenses to successfully stop current mass robocallers and more sophisticated robocallers in the future which may try to overcome simpler defenses. However, we keep AI equipped attackers out of our threat model. Attacks where the attacker is equipped with AI are not common in the phone fraud ecosystem. However, with the availability of tools like Google Duplex, attackers can craft AI equipped attacks where robocallers make a natural conversation with the other party, and fool the VA by pretending to be a human caller. Since RobocallGuardPlus randomly asks a question from a preset question pool, such AI equipped attackers can use NLP based models such BERT [132], Infersent etc to map a question to an encoding. Once they

determine which question has been asked, they can provide a preset appropriate answer to the question. Once robocallers start interacting like humans and go to the extent where even humans have difficulty identifying between a robocaller and a human caller, it is going to be more difficult to stop them.

An important feature of RobocallGuard is that it can also stop unwanted live calls that come from human callers such as telemarketers. However it can not stop targeted calls from unwanted human callers. Conversely, RobocallGuardPlus is effective against targeted robocallers but does not protect against unwanted human callers.

The user studies were conducted with a limited number of people for RobocallGuard and RobocallGuardPlus. Moreover, the sample was limited to college students and their families and the age range of most people were between 20 to 35. A more comprehensive user study with a broader population is needed to get a better perspective of the systems.

As robocallers become smarter and AI equipped with Duplex like system, attempting to stop them might ultimately prove difficult . However, it is important to remember that with each added step, cost is increased for the robocaller trying to evade a defense system. Robocallers aim at making cheap mass calls so that they can reach a large number of targets. As security researchers, our goal is to add additional cost to the robocallers so that it would become economically unattractive. We believe this dissertation helps with achieving that goal.

7.3 Future Work

The ideas and systems developed in this dissertation can be applied and extended to new research. This sections describes several avenues of research that build on the work presented here to study and address remaining and new challenges.

Currently RobocallGuardPlus chooses a random question from a predefined set of questions. RobocallGuardPlus can be improved to ask dynamic questions instead of static questions. RobocallGuardPlus can be thought of like a chatbot which makes conversa-

tion with the caller. The responses by RobocallGuardPlus can be generated using natural language generation. Such a method would further improve the robustness since what RobocallGuardPlus says would depend on the caller's response. In that case, the flow of the conversation would be more natural which would further enhance the user experience. Furthermore, in the future, new ways need to be explored to block unwanted calls that come from live sources [129].

7.4 Concluding Remarks

This dissertation explored ways to bring trust back to the telephony channel. We discussed the threats posed by unwanted callers and explored various defenses to combat current and future robocallers. We investigated currently deployed defenses and we used insights gained from this process to combat more sophisticated robocallers. There are three important contributions. First we explore phone blacklists and show how effective they are in the current scenario. Next we introduced the notion of a virtual assistant aimed with filtering out unwanted calls. To this end, we developed a Smartphone based app named RobocallGuard which can pick up calls from unknown callers on behalf of the user and filter out unwanted calls. The user study we conducted showed that most users believe that such a virtual assistant like system is beneficial to them. Finally, we expand our threat model and introduce RobocallGuardPlus which can effectively block mass, targeted and evasive robocalls. RobocallGuardPlus engages in a natural conversation with the caller and based on the responses provided by the caller to determine if the caller is a robocaller or a human. Such a system is capable to stopping sophisticated robocallers that might emerge in the near future.

This dissertation explores defenses against current robocallers and also investigates defenses against potential future robocallers. As in other areas of cybersecurity, we believe that as current defenses get adopted by more users, in the future robocallers are going to employ more advanced techniques to evade such defenses. In anticipation of such evasion

and to mitigate them, we explored voice assistant mediated interaction to handle incoming calls. We hope that by pursuing the directions set forth by this dissertation, the threats posed by current and future robocallers can be mitigated which will lead to a more trustworthy telephony system. We further hope that the approaches and techniques proposed in this dissertation will generalize to secure other voice communications that might become common in the future as voice is used to control IoT devices and for non-telephony human communication and interaction (e.g., Audio Social Media).

REFERENCES

[1] *How does a robocall work?* https://www.consumer.ftc.gov/articles/0381-how-does-robocall-work-infographic, Accessed: 2020-09-26.

[2] G. Ollmann, *The phishing guide [online]. next generation security software ltd*, 2004.

[3] H. Mustafa, W. Xu, A. R. Sadeghi, and S. Schulz, "You can call but you can't hide: Detecting caller id spoofing attacks," in *2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, IEEE, 2014, pp. 168–179.

[4] *The robocall rebellion*, https://www.nytimes.com/2021/07/28/opinion/the-robocall-rebellion.html, Accessed: 2021-08-01.

[5] *The robocall crisis will never be totally fixed*, https://www.wired.com/story/robocalls-spam-fix-stir-shaken/, Accessed: 2020-09-26.

[6] N. Miramirkhani, O. Starov, and N. Nikiforakis, "Dial one for scam: A large-scale analysis of technical support scams," *arXiv preprint arXiv:1607.06891*, 2016.

[7] *Experts estimate 2015 fraud losses at 38.1 billion (usd)*, https://www.pipelinepub.com/news/global-telecom-fraud-losses-down-significantly, Accessed: 2020-09-26.

[8] P. Hoath, "Telecoms fraud, the gory details," *Computer Fraud & Security*, vol. 1998, no. 1, pp. 10–14, 1998.

[9] H. Tu, A. Doupé, Z. Zhao, and G.-J. Ahn, "Sok: Everyone hates robocalls: A survey of techniques against telephone spam," in *2016 IEEE Symposium on Security and Privacy (SP)*, IEEE, 2016, pp. 320–338.

[10] *Complying with the telemarketing sales rules*, https://www.ftc.gov/tips-advice/business-center/guidance/complying-telemarketing-sales-rule, Accessed: 2020-09-26.

[11] J. LaCour, "Vishing campaign steals card data from customers of dozens of banks," 2014.

[12] M. Collier and D. Endler, *Hacking Exposed Unified Communications & VoIP Security Secrets & Solutions*. McGraw-Hill Osborne Media, 2013.

[13] M. Sahin, A. Francillon, P. Gupta, and M. Ahamad, "Sok: Fraud in telephony networks," in *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*, IEEE, 2017, pp. 235–250.

[14] *Tech support scams*, https://www.ftc.gov/tips-advice/business-center/small-businesses/cybersecurity/tech-support-scams, Accessed: 2020-09-26.

[15] J. Isacenkova, O. Thonnard, A. Costin, A. Francillon, and D. Balzarotti, "Inside the scam jungle: A closer look at 419 scam email operations," *EURASIP Journal on Information Security*, vol. 2014, no. 1, p. 4, 2014.

[16] *The free cruise offer: Scam or legit?* https://www.cruisecritic.com/articles.cfm?ID=1185, Accessed: 2020-09-26.

[17] N. A. Syed, N. Feamster, A. Gray, and S. Krasser, "Snare: Spatio-temporal network-level automatic reputation engine," *Georgia Institute of Technology-CSE Technical Reports-GT-CSE-08-02, Tech. Rep.*, 2008.

[18] A. Ramachandran and N. Feamster, "Understanding the network-level behavior of spammers," in *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, 2006, pp. 291–302.

[19] D. Cook, J. Hartnett, K. Manderson, and J. Scanlan, "Catching spam before it arrives: Domain specific dynamic blacklists," in *Proceedings of the 2006 Australasian workshops on Grid computing and e-research-Volume 54*, 2006, pp. 193–202.

[20] C. J. Dietrich and C. Rossow, "Empirical research of ip blacklists," in *ISSE 2008 Securing Electronic Business Processes*, Springer, 2009, pp. 163–171.

[21] *Hiya*, https://www.hiya.com/, Accessed: 2020-09-26.

[22] *Truecaller*, https://www.truecaller.com/, Accessed: 2020-09-26.

[23] *Youmail*, https://www.youmail.com/, Accessed: 2020-09-26.

[24] *Phone by google – caller id and spam protection*, https://play.google.com/store/apps/details?id=com.google.android.dialer, Accessed: 2020-09-26.

[25] *Ftc*, https://www.ftc.gov, Accessed: 2020-09-26.

[26] P. Gupta, B. Srinivasan, V. Balasubramaniyan, and M. Ahamad, "Phoneypot: Data-driven understanding of telephony threats.," in *NDSS*, vol. 107, 2015, p. 108.

[27] *"neighbor spoofing" is a common type of phone scam*, https://www.bbb.org/article/news-releases/16670-a-new-kind-of-phone-scam-neighbor-spoofing, Accessed: 2020-09-26.

[28] *Strike-force*, https://advocacy.consumerreports.org/press_release/phone-industry-robocall-strike-force-announces-plans-for-tackling-unwanted-calls/, Accessed: 2020-09-26.

[29] *Combating spoofed robocalls with caller id authentication*, https://www.fcc.gov/call-authentication, Accessed: 2020-09-26.

[30] *Secure telephone identity revisited*, https://tools.ietf.org/wg/stir/, Accessed: 2020-09-26.

[31] M. Cohen, E. Finkelman, E. Garr, and B. Moyles, *Call distribution techniques*, US Patent 9,584,658, Feb. 2017.

[32] I. N. Sherman, J. D. Bowers, K. McNamara Jr, J. E. Gilbert, J. Ruiz, and P. Traynor, "Are you going to answer that? measuring user responses to anti-robocall application indicators,"

[33] F. Maggi, "Are the con artists back? a preliminary analysis of modern phone frauds," in *2010 10th IEEE International Conference on Computer and Information Technology*, IEEE, 2010, pp. 824–831.

[34] A. Costin, J. Isacenkova, M. Balduzzi, A. Francillon, and D. Balzarotti, "The role of phone numbers in understanding cyber-crime schemes," in *2013 Eleventh Annual Conference on Privacy, Security and Trust*, IEEE, 2013, pp. 213–220.

[35] B. Srinivasan, A. Kountouras, N. Miramirkhani, M. Alam, N. Nikiforakis, M. Antonakakis, and M. Ahamad, "Exposing search and advertisement abuse tactics and infrastructure of technical support scammers," in *Proceedings of the 2018 World Wide Web Conference*, 2018, pp. 319–328.

[36] M. Bidgoli and J. Grossklags, "Hello. this is the irs calling.: A case study on scams, extortion, impersonation, and phone spoofing," in *2017 APWG Symposium on Electronic Crime Research (eCrime)*, IEEE, 2017, pp. 57–69.

[37] *Old st. trick: Naughty nomorobo traps telemarketers*, https://www.innovateli.com/old-st-trick-naughty-nomorobo-traps-telemarketers/, Accessed: 2020-09-26.

[38] S. Chiappetta, C. Mazzariello, R. Presta, and S. P. Romano, "An anomaly-based approach to the analysis of the social behavior of voip users," *Computer Networks*, vol. 57, no. 6, pp. 1545–1559, 2013.

[39] A. Marzuoli, H. A. Kingravi, D. Dewey, A. Dallas, T. Calhoun, T. Nelms, and R. Pienta, "Call me: Gathering threat intelligence on telephony scams to detect fraud," *Black Hat*, 2016.

[40] N. Jiang, Y. Jin, A. Skudlark, W.-L. Hsu, G. Jacobson, S. Prakasam, and Z.-L. Zhang, "Isolating and analyzing fraud activities in a large cellular network via voice call graph analysis," in *Proceedings of the 10th international conference on Mobile systems, applications, and services*, 2012, pp. 253–266.

[41] V. S. Tseng, J.-C. Ying, C.-W. Huang, Y. Kao, and K.-T. Chen, "Fraudetector: A graph-mining-based framework for fraudulent phone call detection," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015, pp. 2157–2166.

[42] Y. Bai, X. Su, and B. Bhargava, "Detection and filtering spam over internet telephony—a user-behavior-aware intermediate-network-based approach," in *2009 IEEE International Conference on Multimedia and Expo*, IEEE, 2009, pp. 726–729.

[43] S. Yardi, D. Romero, G. Schoenebeck, *et al.*, "Detecting spam in a twitter network," *First monday*, 2010.

[44] V. Balasubramaniyan, M. Ahamad, and H. Park, "Callrank: Combating spit using call duration, social networks and global reputation.," in *CEAS*, 2007.

[45] *Att*, https://www.att.com/features/security-apps.html, Accessed: 2020-09-26.

[46] *Verizon*, https://www.verizon.com/support/residential/homephone/callingfeatures/stop-unwanted-calls, Accessed: 2020-09-26.

[47] B. Srinivasan, P. Gupta, M. Antonakakis, and M. Ahamad, "Understanding cross-channel abuse with sms-spam support infrastructure attribution," in *European Symposium on Research in Computer Security*, Springer, 2016, pp. 3–26.

[48] H. Li, X. Xu, C. Liu, T. Ren, K. Wu, X. Cao, W. Zhang, Y. Yu, and D. Song, "A machine learning approach to prevent malicious calls over telephony networks," in *2018 IEEE Symposium on Security and Privacy (SP)*, IEEE, 2018, pp. 53–69.

[49] S. Pandit, R. Perdisci, M. Ahamad, and P. Gupta, "Towards measuring the effectiveness of telephony blacklists.," in *NDSS*, 2018.

[50] *Neighbor scam grows dramatically in 2018; scammers evolve spoofing tactics*, https://hiya.com/blog/2018/05/23/neighbor-scam-moves-on-to-spoofing-just-area-codes/, Accessed: 2020-09-26.

[51] B. Reaves, L. Blue, H. Abdullah, L. Vargas, P. Traynor, and T. Shrimpton, "Authenticall: Efficient identity and content authentication for phone calls," in *26th {USENIX} Security Symposium ({USENIX} Security 17)*, 2017, pp. 575–592.

[52] B. Reaves, L. Blue, and P. Traynor, "Authloop: End-to-end cryptographic authentication for telephony over voice channels," in *25th {USENIX} Security Symposium ({USENIX} Security 16)*, 2016, pp. 963–978.

[53] H. Tu, A. Doupé, Z. Zhao, and G.-J. Ahn, "Toward authenticated caller id transmission: The need for a standardized authentication scheme in q. 731.3 calling line identification presentation," in *2016 ITU Kaleidoscope: ICTs for a Sustainable World (ITU WT)*, IEEE, 2016, pp. 1–8.

[54] *Trustid*, https://www.trustid.com/, Accessed: 2020-09-26.

[55] V. A. Balasubramaniyan, A. Poonawalla, M. Ahamad, M. T. Hunter, and P. Traynor, "Pindr0p: Using single-ended audio features to determine call provenance," in *Proceedings of the 17th ACM conference on Computer and communications security*, 2010, pp. 109–120.

[56] *Why we're still years away from a robocall-free future*, https://www.cnn.com/2019/04/10/perspectives/stop-robocalls-shaken-stir/index.html, Accessed: 2020-09-26.

[57] *Telephone consumer protection act 47*, https://transition.fcc.gov/cgb/policy/TCPA-Rules.pdf, Accessed: 2020-09-26.

[58] *Truth in caller id act of 2009*, https://www.congress.gov/111/plaws/publ331/PLAW-111publ331.pdf, Accessed: 2020-09-26.

[59] *National do not call registry*, https://www.donotcall.gov/, Accessed: 2020-09-26.

[60] Y. Soupionis and D. Gritzalis, "Audio captcha: Existing solutions assessment and a new implementation for voip telephony," *Computers & Security*, vol. 29, no. 5, pp. 603–618, 2010.

[61] H. Meutzner, S. Gupta, V.-H. Nguyen, T. Holz, and D. Kolossa, "Toward improved audio captchas based on auditory perception and language understanding," *ACM Transactions on Privacy and Security (TOPS)*, vol. 19, no. 4, pp. 1–31, 2016.

[62] J. Tam, J. Simsa, S. Hyde, and L. V. Ahn, "Breaking audio captchas," in *Advances in Neural Information Processing Systems*, 2008, pp. 1625–1632.

[63] E. Bursztein and S. Bethard, "Decaptcha: Breaking 75% of ebay audio captchas," in *Proceedings of the 3rd USENIX conference on Offensive technologies*, USENIX Association, vol. 1, 2009, p. 8.

[64] E. Bursztein, R. Beauxis, H. Paskov, D. Perito, C. Fabry, and J. Mitchell, "The failure of noise-based non-continuous audio captchas," in *2011 IEEE symposium on security and privacy*, IEEE Computer Society, 2011, pp. 19–31.

[65] S. Li, S. A. H. Shah, M. A. U. Khan, S. A. Khayam, A.-R. Sadeghi, and R. Schmitz, "Breaking e-banking captchas," in *Proceedings of the 26th Annual Computer Security Applications Conference*, 2010, pp. 171–180.

[66] S. Solanki, G. Krishnan, V. Sampath, and J. Polakis, "In (cyber) space bots can hear you speak: Breaking audio captchas using ots speech recognition," in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, 2017, pp. 69–80.

[67] V. Fanelle, S. Karimi, A. Shah, B. Subramanian, and S. Das, "Blind and human: Exploring more usable audio captcha designs," in *Sixteenth Symposium on Usable Privacy and Security (SOUPS 2020)*, 2020, pp. 111–125.

[68] N. Mrkšić, D. O. Séaghdha, T.-H. Wen, B. Thomson, and S. Young, "Neural belief tracker: Data-driven dialogue state tracking," *arXiv preprint arXiv:1606.03777*, 2016.

[69] Z. Yan, N. Duan, J. Bao, P. Chen, M. Zhou, Z. Li, and J. Zhou, "Docchat: An information retrieval approach for chatbot engines using unstructured documents," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2016, pp. 516–525.

[70] L. Zhou, J. Gao, D. Li, and H.-Y. Shum, "The design and implementation of xiaoice, an empathetic social chatbot," *Computational Linguistics*, vol. 46, no. 1, pp. 53–93, 2020.

[71] D. G. Bobrow, R. M. Kaplan, M. Kay, D. A. Norman, H. Thompson, and T. Winograd, "Gus, a frame-driven dialog system," *Artificial intelligence*, vol. 8, no. 2, pp. 155–173, 1977.

[72] D. Suendermann, K. Evanini, J. Liscombe, P. Hunter, K. Dayanidhi, and R. Pieraccini, "From rule-based to statistical grammars: Continuous improvement of large-scale spoken dialog systems," in *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, IEEE, 2009, pp. 4713–4716.

[73] J. Weizenbaum, "Eliza—a computer program for the study of natural language communication between man and machine," *Communications of the ACM*, vol. 9, no. 1, pp. 36–45, 1966.

[74] C.-W. Liu, R. Lowe, I. V. Serban, M. Noseworthy, L. Charlin, and J. Pineau, "How not to evaluate your dialogue system: An empirical study of unsupervised evalua-

tion metrics for dialogue response generation," *arXiv preprint arXiv:1603.08023*, 2016.

[75] K. Gopalakrishnan, B. Hedayatnia, Q. Chen, A. Gottardi, S. Kwatra, A. Venkatesh, R. Gabriel, D. Hakkani-Tür, and A. A. AI, "Topical-chat: Towards knowledge-grounded open-domain conversations.," in *INTERSPEECH*, 2019, pp. 1891–1895.

[76] I. V. Serban, R. Lowe, P. Henderson, L. Charlin, and J. Pineau, "A survey of available corpora for building data-driven dialogue systems," *arXiv preprint arXiv:1512.05742*, 2015.

[77] S. H. Kimball, T. Levy, H. Venturelli, and S. Miller, "Interactive voice recognition communication in electoral politics: Exploratory metadata analysis," *American Behavioral Scientist*, vol. 58, no. 9, pp. 1236–1245, 2014.

[78] H. Tu, A. Doupé, Z. Zhao, and G.-J. Ahn, "Users really do answer telephone scams," in *28th {USENIX} Security Symposium ({USENIX} Security 19)*, 2019, pp. 1327–1340.

[79] *Pindrop*, https://www.pindrop.com, Accessed: 2020-09-26.

[80] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, *et al.*, "The kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*, IEEE Signal Processing Society, 2011.

[81] *Whitepages*, http://www.whitepages.com, Accessed: 2020-09-26.

[82] P. Prakash, M. Kumar, R. R. Kompella, and M. Gupta, "Phishnet: Predictive blacklisting to detect phishing attacks," in *2010 Proceedings IEEE INFOCOM*, IEEE, 2010, pp. 1–5.

[83] A. Ramachandran, D. Dagon, and N. Feamster, "Can dns-based blacklists keep up with bots?" In *CEAS*, 2006.

[84] M. Antonakakis, R. Perdisci, D. Dagon, W. Lee, and N. Feamster, "Building a dynamic reputation system for dns.," in *USENIX security symposium*, 2010, pp. 273–290.

[85] M. Felegyhazi, C. Kreibich, and V. Paxson, "On the potential of proactive domain blacklisting.," *LEET*, vol. 10, pp. 6–6, 2010.

[86] *At&t call protect*, https://www.att.com/offers/call-protect.html, Accessed: 2020-09-26.

[87] *Google android dialer*, https://play.google.com/store/apps/details?id=com.google.android.dialer, Accessed: 2020-09-26.

[88] D. M. J. Tax, "One-class classification: Concept learning in the absence of counter-examples.," 2002.

[89] M. M. Moya and D. R. Hush, "Network constraints and multi-objective optimization for one-class classification," *Neural networks*, vol. 9, no. 3, pp. 463–474, 1996.

[90] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American society for information science*, vol. 41, no. 6, pp. 391–407, 1990.

[91] J. Ramos *et al.*, "Using tf-idf to determine word relevance in document queries," in *Proceedings of the first instructional conference on machine learning*, Citeseer, vol. 242, 2003, pp. 29–48.

[92] *Singular value decomposition*, https://en.wikipedia.org/wiki/Singularvaluedecomposition., Accessed: 2020-09-26.

[93] D. Mimno, H. Wallach, E. Talley, M. Leenders, and A. McCallum, "Optimizing semantic coherence in topic models," in *Proceedings of the 2011 conference on empirical methods in natural language processing*, 2011, pp. 262–272.

[94] R. Dantu and P. Kolan, "Detecting spam in voip networks.," *SRUTI*, vol. 5, pp. 5–5, 2005.

[95] D. Endler and M. Collier, *Hacking exposed voip: Voice over ip security secrets & solutions. 2006*.

[96] *Yellowpages*, https://www.yellowpages.com/, Accessed: 2020-09-26.

[97] *Fcc seeks reliable call authentication system*, https://www.fcc.gov/document/fcc-seeks-reliable-call-authentication-system, Accessed: 2020-09-26.

[98] S. E. Griffin and C. C. Rackley, "Vishing," in *Proceedings of the 5th annual conference on Information security curriculum development*, 2008, pp. 33–35.

[99] *Google duplex*, https://ai.googleblog.com/2018/05/duplex-ai-system-for-natural-conversation.html, Accessed: 2020-09-26.

[100] *Google pixel*, https://support.google.com/phoneapp/answer/9118387?hl=en, Accessed: 2020-09-26.

[101] *Snowboy*, https://snowboy.kitt.ai/, Accessed: 2020-09-26.

[102] *Cmu pocketsphinx*, https://cmusphinx.github.io/, Accessed: 2020-09-26.

[103] *Honk*, https://github.com/castorini/honk, Accessed: 2020-09-26.

[104] *Webrtc vad*, https://github.com/wiseman/py-webrtcvad, Accessed: 2020-09-26.

[105] *Google speech*, https://cloud.google.com/speech-to-text/, Accessed: 2020-09-26.

[106] *Deep speech*, https://github.com/mozilla/DeepSpeech, Accessed: 2020-09-26.

[107] *Android*, https://developer.android.com/guide/topics/media/mediaplayer.html, Accessed: 2020-09-26.

[108] *Firebase*, https://firebase.google.com/docs/cloud-messaging, Accessed: 2020-09-26.

[109] L. Faulkner, "Beyond the five-user assumption: Benefits of increased sample sizes in usability testing," *Behavior Research Methods, Instruments, & Computers*, vol. 35, no. 3, pp. 379–383, 2003.

[110] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise.," in *kdd*, vol. 96, 1996, pp. 226–231.

[111] *Twilio*, https://www.twilio.com/, Accessed: 2020-09-26.

[112] *Spoofcard*, https://www.spoofcard.com/apps, Accessed: 2020-09-26.

[113] I. N. Sherman, J. Bowers, K. McNamara Jr, J. E. Gilbert, J. Ruiz, and P. Traynor, "Are you going to answer that? measuring user responses to anti-robocall application indicators.," in *NDSS*, 2020.

[114] A. Sciuto, A. Saini, J. Forlizzi, and J. I. Hong, "" hey alexa, what's up?" a mixed-methods studies of in-home conversational agent usage," in *Proceedings of the 2018 Designing Interactive Systems Conference*, 2018, pp. 857–868.

[115] P. Shah, D. Hakkani-Tür, G. Tür, A. Rastogi, A. Bapna, N. Nayak, and L. Heck, "Building a conversational agent overnight with dialogue self-play," *arXiv preprint arXiv:1801.04871*, 2018.

[116] C. Khatri, A. Venkatesh, B. Hedayatnia, R. Gabriel, A. Ram, and R. Prasad, "Alexa prize—state of the art in conversational ai," *AI Magazine*, vol. 39, no. 3, pp. 40–55, 2018.

[117] L. Clark, N. Pantidi, O. Cooney, P. Doyle, D. Garaialde, J. Edwards, B. Spillane, E. Gilmartin, C. Murad, C. Munteanu, *et al.*, "What makes a good conversation? challenges in designing truly conversational agents," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 2019, pp. 1–12.

[118] T.-H. K. Huang, W. S. Lasecki, A. Azaria, and J. P. Bigham, "" is there anything else i can help you with?" challenges in deploying an on-demand crowd-powered conversational agent," in *Fourth AAAI Conference on Human Computation and Crowdsourcing*, 2016.

[119] A. Wald, "Sequential tests of statistical hypotheses," *The annals of mathematical statistics*, vol. 16, no. 2, pp. 117–186, 1945.

[120] M. Allahyari, S. Pouriyeh, M. Assefi, S. Safaei, E. D. Trippe, J. B. Gutierrez, and K. Kochut, "Text summarization techniques: A brief survey," *arXiv preprint arXiv:1707.02268*, 2017.

[121] M. Gambhir and V. Gupta, "Recent automatic text summarization techniques: A survey," *Artificial Intelligence Review*, vol. 47, no. 1, pp. 1–66, 2017.

[122] Y. Liu and M. Lapata, "Text summarization with pretrained encoders," *arXiv preprint arXiv:1908.08345*, 2019.

[123] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[124] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[125] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *arXiv preprint arXiv:1802.05365*, 2018.

[126] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.

[127] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes, "Supervised learning of universal sentence representations from natural language inference data," *arXiv preprint arXiv:1705.02364*, 2017.

[128] J. Chen, Y. Wu, and D. Yang, "Semi-supervised models via data augmentation-for classifying interactive affective responses," *arXiv preprint arXiv:2004.10972*, 2020.

[129] M. Sahin, M. Relieu, and A. Francillon, "Using chatbots against voice spam: Analyzing lenny's effectiveness," in *Thirteenth Symposium on Usable Privacy and Security ({SOUPS} 2017)*, 2017, pp. 319–337.

[130] *Spacy*, https://spacy.io/, Accessed: 2020-09-26.

[131] V. Keselj, *Speech and language processing daniel jurafsky and james h. martin (stanford university and university of colorado at boulder) pearson prentice hall, 2009, xxxi+ 988 pp; hardbound, isbn 978-0-13-187321-6*, 2009.

[132] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.