Characterizing and Detecting Online Deception via Data-Driven Methods

Hang Hu

Dissertation submitted to the Faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Computer Science and Applications

Gang Wang, Co-chair
Danfeng (Daphne) Yao, Co-chair
Wenjing Lou
Yaling Yang
Yuan Tian

May 8, 2020
Blacksburg, Virginia

Keywords: Spear-Phishing, Email Spoofing, Email Tracking, Domain Squatting, Password Reuse, Voice Personal Assistant
Copyright 2020, Hang Hu
In recent years, online deception has become a major threat to information security. Online deception that caused significant consequences is usually spear phishing. Spear-phishing emails come in a very small volume, target a small number of audiences, sometimes impersonate a trusted entity and use very specific content to redirect targets to a phishing website, where the attacker tricks targets sharing their credentials.

In this thesis, we aim at measuring the entire process. Starting from phishing emails, we examine anti-spoofing protocols, analyze email services’ policies and warnings towards spoofing emails, and measure the email tracking ecosystem. With phishing websites, we implement a powerful tool to detect domain name impersonation and detect phishing pages using dynamic and static analysis. We also analyze credential sharing on phishing websites, and measure what happens after victims share their credentials. Finally, we discuss potential phishing and privacy concerns on new platforms such as Alexa and Google Assistant.

In the first part of this thesis (Chapter 3), we focus on measuring how email providers detect and handle forged emails. We also try to understand how forged emails can reach user inboxes by deliberately composing emails. Finally, we check how email providers warn users about forged emails. In the second part (Chapter 4), we measure the adoption of anti-spoofing protocols and seek to understand the reasons behind the low adoption rates. In the third part of this thesis (Chapter 5), we observe that a lot of phishing emails use email tracking techniques to track targets. We collect a large dataset of email messages using disposable email services and measure the landscape of email tracking. In the fourth part of this thesis (Chapter 6), we move on to phishing websites. We implement a powerful tool to detect squatting domains and train a machine learning model to classify phishing websites. In the fifth part (Chapter 7), we focus on the credential leaks. More specifically, we measure what happens after the targets’ credentials are leaked. We monitor and measure the potential post-phishing exploiting activities. Finally, with new voice platforms such as Alexa becoming more and more popular, we wonder if new phishing and privacy concerns emerge with new platforms. In this part (Chapter 8), we systematically assess the attack surfaces by measuring sensitive applications on voice assistant systems.

My thesis measures important parts of the complete process of online deception. With deeper understandings of phishing attacks, more complete and effective defense mechanisms can be developed to mitigate attacks in various dimensions.
In recent years, online deception becomes a major threat to information security. The most common form of online deception starts with a phishing email, then redirects targets to a phishing website where the attacker tricks targets sharing their credentials. General phishing emails are relatively easy to recognize from both the target’s and the defender’s perspective. They are usually from strange addresses, the content is usually very general and they come in a large volume. However, Online deception that caused significant consequences is usually spear phishing. Spear-phishing emails come in a very small volume, target a small number of audiences, sometimes impersonate a trusted entity and use very specific content to redirect targets to a phishing website, where the attacker tricks targets sharing their credentials. Sometimes, attackers use domain impersonation techniques to make the phishing website even more convincing.

In this thesis, we measure the entire process. Starting from phishing emails, we examine anti-spoofing protocols, analyze email services’ policies and warnings towards spoofing emails, and measure the email tracking ecosystem. With phishing websites, we implement a tool to detect domain name impersonation and detect phishing pages using dynamic and static analysis. We also studied credential sharing on phishing websites. We measure what happens after targets share their credentials. Finally, we analyze potential phishing and privacy concerns on new platforms such as Alexa and Google Assistant.
Dedication

Dedicated to my parents Zongzhi Hu and Xiufeng Yang, and my husband Daniel Jude, for their support, motivation, encouragement, and love.
Acknowledgments

This Ph.D. journey will not even be possible without help from many people I met. So with deep gratitude, I want to use this section to thank those awesome people.

First, I would like to thank my co-advisors Dr. Gang Wang and Dr. Danfeng (Daphne) Yao. I conducted most of my research works with my co-advisors. Dr. Wang joined VT the same year as I did in 2016 as a fresh new assistant professor. I was the first few students that started to do fulltime projects with him. Dr. Wang spent a lot of time and energy training me to be a good Ph.D. student. He guided me on how to conduct researches and projects efficiently. Dr. Wang was also very patient and encouraging. The first research project I led was rejected four times before it was finally accepted, spanning over a year. The process of rebuttal, rejection, revision, and resubmission was frustrating for a new Ph.D. student entering the academic world. If it was not for Dr. Wang’s insight and persistence, the work would never improve and make it to a top conference. In the following years working with Dr. Wang, I learnt a lot from him. Also Dr. Yao provided a lot of comments and teaching on my presentations and thesis organizations. I would also like to thank my Ph.D. committee members: Dr. Wenjing Lou, Dr. Yaling Yang, and Dr. Yuan Tian for their insightful comments and feedback on my thesis various stages of my Ph.D.

I want to thank my husband Daniel Jude. Without his emotional support, it’s impossible for me to finish my PhD. With the busy schedule of projects and frustration with submission rejection, the PhD journey was tough for me from time to time. From what I know, being depressed during PhD study is pretty normal in recent years. Daniel taught me to love myself and debate with the constant negative thoughts going on in my mind. Gradually, I became more satisfied with my life and happier with everything in general. I am truly grateful that I have met Daniel. He made me a much happier version of myself. I am also very grateful for my parents Zongzhi Hu and Xiufeng Yang. They gave me the best environment to grow up in. They loved each other and showed how to be responsible, reliable and hardworking.

During my studies in Blacksburg, I made a lot of friends. I felt extremely lucky to have met them, especially two people Hao Zou and Yongwen Tan. Hao is a physics Ph.D. student at VT, and Yongwen is a visiting Ph.D. student in mechanical engineering. Hao is a string theory researcher, neither academic nor industrial opportunities are very available in this field. However, he truly enjoys what he is doing. His optimism affects me and makes me feel encouraged in my own career path. I met Yongwen while going on a camping trip with friends. Ever since that, we became friends. For the short two years of Yongwen staying in Blacksburg, He would make dinner and invite me three times a week, and we would talk about everything that’s going on in our life. I am also extremely fortunate to have worked
with a wonderful group of labmates. A shout-out to labmates at Virginia Tech: Steve Jan, Peng Peng, Chao Xu, Limin Yang, and Qingying Hao. Studying in a foreign country can be lonely sometimes, those friends made the journey much easier.
# Contents

List of Figures xiii

List of Tables xvi

1 Introduction 1

1.1 Email Impersonation ........................................... 1
1.2 Disposable Email Service ...................................... 2
1.3 Email Tracking .................................................. 3
1.4 Web Phishing ..................................................... 3
1.5 Credential Sharing ............................................... 4
1.6 Voice Personal Assistant ....................................... 4
1.7 Contributions .................................................... 5

2 Background 7

2.1 SMTP and Email Spoofing ...................................... 7
   2.1.1 Email Authentication ....................................... 7
   2.1.2 The Low Adoption Rates of Anti-spoofing Protocols ...... 8
2.2 Disposable Email Services ..................................... 9
2.3 Email Tracking .................................................. 10
2.4 Phishing Web Pages ............................................. 11
2.5 Voice Personal Assistant ....................................... 13

3 Email Spoofing Attacks 15

3.1 Introduction ..................................................... 15
3.2 Research Questions and Methodology ......................... 17
3.3 Adoption of SMTP Extensions ................................ 19
3.4 End-to-End Spoofing Experiments ............................................. 21
  3.4.1 Experiment Setup .......................................................... 21
  3.4.2 Experiment Parameters .................................................... 22
3.5 Spoofing Experiment Results .................................................. 23
  3.5.1 Authentication Mechanisms .............................................. 23
  3.5.2 Decisions on Forged Emails ............................................ 24
  3.5.3 Email Clients and Security Indicators ................................. 28
  3.5.4 Misleading UI Elements .................................................. 29
3.6 Effectiveness of Security Indicators ........................................ 30
  3.6.1 Experiment Methodology .................................................. 31
  3.6.2 Experiment Results ....................................................... 33
3.7 Discussion .............................................................................. 35
  3.7.1 Implications of Our Results .............................................. 36
  3.7.2 UI Updates from Email Services ........................................ 37
  3.7.3 Open Questions & Limitations ........................................... 37
3.8 Related Work ......................................................................... 38
3.9 Conclusion .............................................................................. 39

4 The Adoption of Anti-Spoofing Protocols in Email Systems ........... 40
  4.1 Introduction .......................................................................... 40
  4.2 User Study Methodology ....................................................... 42
  4.3 User Study Results ............................................................... 44
    4.3.1 Technical Defects of the Protocols .................................. 44
    4.3.2 A Lack of Critical Mass ................................................ 46
    4.3.3 Benefits Not Significantly Overweight Costs ....................... 46
    4.3.4 Deployment Difficulties in Practice .................................. 48
    4.3.5 Risks of Breaking the Existing System ............................... 49
    4.3.6 Solutions Moving Forward ............................................. 50
  4.4 Discussion .............................................................................. 51
6.1 Acknowledgement .................................................. 79
6.2 Introduction ....................................................... 79
6.3 Research Questions ................................................ 81
6.4 Measurement Methodology ........................................ 82
  6.4.1 Squatting Detection ........................................... 82
  6.4.2 Web Crawling .................................................. 86
6.5 Characterizing Evasions .......................................... 88
  6.5.1 Ground Truth Phishing Pages ................................ 88
  6.5.2 Evasion Measurement ......................................... 90
6.6 Machine-Learning Detection .................................... 92
  6.6.1 Feature Engineering .......................................... 92
  6.6.2 Feature Embedding and Training .............................. 94
  6.6.3 Ground-Truth Evaluation .................................... 95
6.7 Squatting Phishing in the Wild .................................. 96
  6.7.1 Detecting Squatting Phishing Pages ......................... 96
  6.7.2 Squatting Types & Case Studies .............................. 99
  6.7.3 Evasion .......................................................... 101
6.8 Discussion .......................................................... 102
6.9 Related Work ....................................................... 103
6.10 Conclusion .......................................................... 104

7 Credential Sharing on Phishing Sites .................................... 106
  7.1 Acknowledgement .................................................. 106
  7.2 Introduction ....................................................... 106
  7.3 Motivations .......................................................... 108
    7.3.1 Methodology Overview ..................................... 108
  7.4 Tool Design & Data Collection .................................. 109
    7.4.1 Measurement Tool ........................................... 109
    7.4.2 Data Collection .............................................. 111
7.5 Client Side Analysis ............................................ 112
  7.5.1 Understanding Phishing Sites .......................... 113
  7.5.2 Client-Side Information Flow .......................... 115
7.6 Server side analysis ............................................ 120
  7.6.1 Collecting Phishing Kits ................................. 120
  7.6.2 Server-side Information Flow .......................... 121
7.7 Post-Phishing Exploitation .................................... 123
  7.7.1 Experiment Setup ....................................... 124
  7.7.2 Activities on Honeypot Accounts ...................... 126
7.8 Discussion ................................................... 127
7.9 Related Work ................................................ 128
7.10 Conclusion .................................................. 129

8 Sensitive Applications of Voice Personal Assistant Systems 130
  8.1 Acknowledgement .......................................... 130
  8.2 Introduction ............................................... 130
  8.3 Problem Definition and Goals ............................ 133
    8.3.1 Threat Model ....................................... 133
    8.3.2 Analysis Goals .................................... 134
    8.3.3 Motivating Examples ................................. 135
  8.4 Experiment Design ......................................... 135
    8.4.1 Data Collection ..................................... 136
    8.4.2 Data Pre-processing ................................. 137
    8.4.3 Data Labeling ...................................... 138
    8.4.4 Experiment 1. Capability Analysis Model .......... 139
    8.4.5 Experiment 2. Sensitivity Analysis Model ........ 140
    8.4.6 Sanity Check for Undocumented Voice Commands .. 142
  8.5 Measurements and Evaluations ............................ 143
    8.5.1 Capability Analysis Model .......................... 144
8.5.2 Sensitivity Analysis Model .................................................. 145
8.5.3 Measuring Security Implications of Skills. ............................ 146
8.5.4 Sanity check evaluation ..................................................... 148
8.6 Discussion ........................................................................... 149
8.7 Related Work ..................................................................... 150
8.8 Conclusion .......................................................................... 150

9 Conclusions and Future Works .................................................. 153

9.1 Conclusion ........................................................................... 153
9.2 Future Works ...................................................................... 154

Bibliography .............................................................................. 155

Appendices ................................................................................. 180

A Spoofing Target Domains ......................................................... 181
B Obfuscated User Identifier ........................................................ 182
C Example of Sensitive and Non-Sensitive Voice Commands. ........ 182
### List of Figures

2.1 Email transmission from Alex to Bob. .................................................. 8
2.2 Two types of disposable email addresses. ............................................. 9
2.3 Phishing attack process. ................................................................. 11
2.4 System model of how Amazon Alexa process a voice command. .......... 13

3.1 The adoption rate of SPF and DMARC among Alexa 1 million domains across three snapshots. ........................................... 19
3.2 The adoption rate as a function of the domains’ Alexa rankings (January 2018). 20
3.3 End-to-end spoofing experiment setup. We use our server E.com to send a forged email to the target email service B.com by spoofing A.com. ........ 22
3.4 The aggregated ratio of emails that reached the user inbox (inbox rate). The legend displays the 3 authentication groups of the receivers. Each subfigure shows the breakdown results for emails with specific configurations. .. 25
3.5 Security indicators on forged emails from 9 email providers. (a)–(e) are for regular forged emails. (f)–(h) only show up when the spoofed sender and the receiver belong to the same provider. (i) only shows up when spoofing an existing contact. .................................................. 27
3.6 Seznam.cz displays a “trusted address” sign on a forged address. ............ 30
3.7 The phishing email screenshot. .......................................................... 32
3.8 The joint impact of demographic factors and security indicators on click rates. 35
3.9 Gmail’s new warning message for same-domain spoofing. ................... 35

4.1 A spoofing email that impersonates the U.S. Citizenship and Immigration Services (USCIS). We acted as the attacker and sent this email to our own account. The email arrived the inbox without triggering any alert. .......... 41
4.2 SPF: SPF test is based on the domain of “Return-Path”, which can be different from the domain that the user sees (the “From” field). .......... 45
4.3 The adoption model for anti-spoofing protocols. For email domains, the cost and benefit changes as more domains adopt the protocol. For non-email domains, the cost and benefit stay constant. .............. 47
5.1 Email classification results. ................................. 63
5.2 Distribution of the HTML image size. ....................... 68
5.3 The HTML image size of invisible remote pixels. ............ 68
5.4 # of tracking URLs under different tracking methods. ....... 68
5.5 Different tracking methods of first-party and third-party trackers. ................................. 71
5.6 # of third-party trackers per sender. ....................... 71
5.7 # of sender domains associated to each tracker. ................ 71
5.8 Tracking methods used by popular (Alexa top 10K) and non-popular sender domains. ................ 74
5.9 Tracking methods used by popular (Alexa top 10K) and non-popular sender domains. ................ 74
5.10 Evasion methods used by popular (Alexa top 10k) and non-popular sender domains. ................ 74
5.11 Type of tracking used by different sender domains. ........... 74

6.1 An example of the internationalized domain name xn--facbook-ts4c.com (homograph), which is displayed as facebook.com in the address bar. .... 80
6.2 # of squatting domain of different squatting types. ............ 83
6.3 Accumulated % of squatting domains from top brands. Brands sorted by # of domains. ....................... 83
6.4 Accumulated % phishing URLs from top brands in PhishTank. ......... 88
6.5 Alexa ranking of phishing URLs in PhishTank. ................ 88
6.6 PhishTank squatting domains distribution. ..................... 88
6.7 An example of page layout obfuscation of phishing pages (paypal). ....... 90
6.8 Average Image hash distance and standard variance for phishing pages of different brands. ....................... 90
6.9 False positive rate vs. true positive rate of different models. ........ 94
6.10 # of verified phishing domains for each brand. ............... 94
6.11 # of squatting phishing domains of different squatting types. .......... 94
6.12 The top 70 brands targeted by squatting phishing pages. ........... 96
6.13 Examples of squatting phishing pages. ....................... 98
6.14 The location of squatting phishing websites. ........................................ 99
6.15 The registration time of squatting phishing domains. ............................ 99
6.16 # of phishing pages within each snapshot. ................................................. 99

7.1 The gap between the time when a URL was blacklisted and the time when our crawler visited the URL ............................................................... 111
7.2 Compromised domains and their hosted phishing pages. .......................... 112
7.3 Geolocation distribution of phishing URLs. ................................................. 114
7.4 Countries of phishing sites and third-party collectors. ............................... 117
7.5 Distribution of fraction of phishing sites that connect to different third-party collectors. On the x-axis, third-party collectors are ranked based on # of phishing sites connected. ............................................................... 118
7.6 CCDF of Number of VirusTotal scanners that flagged the given URL as malicious. The majority of the third-party collectors are already flagged by VirusTotal scanners. ............................................................... 118
7.7 Registration time of phishing domains and third-party collector domains. Third-party collector domains have a similar distribution with phishing domains. ............................................................... 118
7.8 Number of server-side collectors per phishing kit. .................................... 122

8.1 Two types of attack against the VPA system. ............................................ 133
8.2 System Overview for the keyword-based approach of finding sensitive and nonsensitive voice commands. ............................................................. 141
8.3 We compare our active learning using margin sampling model’s performance with four different baseline approaches including-(1) Base RNN; where we use RNN network structure for building the machine learning model, (2) Base CNN; where we use CNN network structure, (3) CNN + data clean; where before training the model, we process the input data according to Section 8.4.2, (4) CNN + ActiveL (Entropy); where we use entropy metric to select unlabeled data to be labeled in each round of active learning approach, (5) CNN + ActiveL (Margin) is our proposed method; where we select the most uncertain unlabeled data to be labeled in each round of active learning approach (sorted based on F1-score). ............................................................. 145

A.1 Examples of misleading UIs (profile photo, email history, namecard). ....... 188
## List of Tables

3.1 SPF/DMARC statistics of Alexa 1 million domains. The data was collected in January 2018. .................................................. 18

3.2 The ratio of emails that reached the inbox (inbox rate). We break down the inbox rate for emails with different configuration parameters (sender IP, the SPF/DKIM/DMARC profile of the sender address, and the email content). 24

3.3 Feature ranking. ............................................................. 27

3.4 Misleading UI elements when the attacker spoofs an existing contact. (*) indicates web interface only. (†) indicates mobile only. .................................. 29

3.5 User study statistics. .......................................................... 34

3.6 User study statistics for different user-agents. ............................. 34

4.1 User study participants: 9 email administrators. U8 requested to conceal the institution type, and thus we keep it as “anonymous”. For each of their email services, we also measured whether the email domain published the DNS authentication records (as the sender) and whether the domain authenticate incoming emails (as the receiver). “✓” means the mail server has adopted SPF/DKIM/DMARC. “✗” means the mail server did has not adopted SPF/DKIM/DMARC. “/” means not applicable. Note that we could not obtain a mail server’s DKIM record from the DNS since the selector information is not public. .................................................. 42

4.2 Technical weaknesses of SPF, DKIM and DMARC. ......................... 43

5.1 The expiration time of disposable emails. We show the expiration time claimed on the website and the actual expiration time obtained through measurements. .................................................. 58

5.2 Statistics of the collected datasets. ............................................. 59

5.3 PII detection accuracy based on ground-truth, and the number of detected PII instances in our dataset. ............................................. 61

5.4 Top 5 sender domains of registration emails, password reset emails and authentication emails. .................................................. 64
5.5 Top 10 categories of the email sender domains for spam and account management emails. .......................................................... 65
5.6 Email tracking detection results. *Tracking party is based on 1.29 million emails that have a sender address. ................................. 68
5.7 Obfuscation methods used in the tracking URLs. .............................. 70
5.8 Top 10 hidden trackers, ranked by the # of trackers that redirect traffic to them. ................................................................. 72
5.9 Top third-party trackers for each type of tracking method. ................. 72
5.10 Top third-party trackers across the full dataset. “●” means the tracker is also a hidden tracker. “○” means the tracker is not a hidden tracker. .......................... 73

6.1 Examples of different types of squatting domains for the facebook brand. .. 83
6.2 Top 5 brands with the most squatting domains. ................................. 84
6.3 Crawling statistics. We measure the redirections to the original website and those to domain marketplaces. ................................ 85
6.4 Top brands with the highest ratio of redirections to their original websites. 85
6.5 Top brands with highest ratio of redirections to domain marketplaces. ...... 86
6.6 Top 8 brands in PhishTank cover 4004 phishing URLs (59.1%). Manual verification shows that 1731 pages are true phishing pages. .......... 89
6.7 String and code obfuscation in phishing pages. ................................ 92
6.8 Classifiers’ performance on ground-truth data. ................................ 95
6.9 Detected and confirmed squatting phishing pages. ............................ 96
6.10 15 example brands and verified phishing pages. ............................... 97
6.11 Phishing pages that adopted evasion techniques. ............................. 101
6.12 Detected squatting phishing pages by popular blacklists. VirusTotal contains 70+ blacklists. ................................................. 101
6.13 The liveness of phishing pages on different dates. ............................. 102
6.14 Selected example phishing domains for 15 different brands. Note that “●” means web page only. “○” means mobile page only. The rest have both web and mobile pages. ........................................ 105

7.1 Dataset summary. ................................................................. 109
7.2 Functions used to obfuscate login credentials. .................................. 112
7.3 Top 10 domains of phishing URLs. .................................................. 113
7.4 Compromised domains that host phishing pages. ............................ 114
7.5 Target sectors and top brands in each sector. ................................. 115
7.6 Data format of credentials sent from the client-side. ......................... 116
7.7 Distribution of third-party collectors. About 95% phishing sites don’t have third-party collectors and they only send credentials to the original hosting domain. .............................................................. 116
7.8 Number of URLs detected by VirusTotal. ........................................ 118
7.9 Top 10 third-party collectors. ......................................................... 119
7.10 Top 5 third-party collectors on the server side. .............................. 123
7.11 Top 5 first-party collectors on the server side. ............................... 123
7.12 Collectors on both client and server side. ....................................... 124
7.13 Account exploitation activities in our honey accounts. .................... 124
8.1 Example skills and their sensitive voice commands. .......................... 135
8.2 Our dataset. ..................................................................................... 136
8.3 Ground-truth data of Amazon Alexa US 2019 for Capability and Sensitivity analysis model. ................................................................. 144
8.4 Overview of the total number of action injection-sensitive, action injection-non sensitive, information retrieval-sensitive, information retrieval-non sensitive voice commands in Amazon Alexa and Google Home. ........... 147
8.5 We use our two models- active learning model & keyword-based model, to identify the total number of action injection-sensitive, action injection-nonsensitive, information retrieval-sensitive, information retrieval-nonsensitive voice commands in 80,129 voice commands from twenty-three different categories of Amazon Alexa US 2019. Inject. means action injection. Retriv. means information retrieval. ................................................................. 151
8.6 Analysis results for 9,096 voice commands from eighteen different categories of Google Home 2019. ......................................................... 152
A.1 Spoofed Sender Domain List. ......................................................... 181
B.1 Functions to obfuscate user identifiers. .......................................... 182
C.1 Sensitive commands from Google Home. .......................... 183
C.2 Sensitive commands from Amazon Alexa. ......................... 184
C.3 Non-sensitive commands from Amazon Alexa. .................... 185
C.4 Non-sensitive commands from Google Home. ................. 187
Chapter 1

Introduction

1.1 Email Impersonation

Despite the recent development of the system and network security, human factors still remain a weak link. As a result, attackers increasingly rely on phishing tactics to breach various target networks [275]. For example, email phishing has involved in nearly half of the 2000+ reported security breaches in recent two years, causing a leakage of billions of user records [288], ransomware outbreaks [207], and even political campaigns [122].

Email spoofing is a critical step in phishing, where the attacker impersonates a trusted entity to gain the victim’s trust. According to the recent report from the Anti-Phishing Working Group (APWG), email spoofing is widely in spear phishing attacks to target employees of various businesses [12]. Unfortunately, today’s email transmission protocol (SMTP) has no built-in mechanism to prevent spoofing [242]. It relies on email providers to implement SMTP extensions such as SPF [173], DKIM [104] and DMARC [204] to authenticate the sender. Since implementing these extensions is voluntary, their adoption rate is far from satisfying. Real-world measurements conducted in 2015 have shown that among Alexa top 1 million domains, 40% have SPF, 1% have DMARC, and even fewer are correctly/strictly configured [114, 131]. As a result, sending spoofing emails is still surprisingly easy today.

The limited server-side protection is likely to put users in a vulnerable position. Since not every sender domain has adopted SPF/DKIM/DMARC, email providers still face key challenges to reliably authenticate all the incoming emails. When an email failed the authentication, it is a “blackbox” process in terms of how email providers handle this email. Would forged emails still be delivered to users? If so, how could users know the email is questionable? Take Gmail for example, Gmail delivers certain forged emails to the inbox and places a security indicator on the sender icon (a red question mark, Figure 3.5(a)). We are curious about how a broader range of email providers handle forged emails, and how much the security indicators actually help to protect users.

Another question is, why email spoofing is still possible after years of efforts spent on the defense. In 2015, two measurement studies [114, 131] show that the adoption rates of anti-spoofing protocols are still low. Among Alexa top 1 million domains, only 40% have adopted SPF and only 1% have DMARC. We repeated the same measurement methodology recently in 2018, and found that the adoption rates were not significantly improved (SPF 44.9%,
Chapter 1. Introduction

DMARC 5.1%). It is not yet clear what causes the slow progress of adopting anti-spoofing solutions.

In this thesis, we describe our efforts and experience in evaluating the real-world defenses against email spoofing. We answer the above questions through empirical end-to-end spoofing measurements, and a user study. First, we conduct measurements on how popular email providers detect and handle forged emails. The key idea is to treat each email provider as a blackbox and vary the input (forged emails) to monitor the output (the receiver’s inbox). Our goal is to understand under what conditions the forged/phishing emails are able to reach the user inbox and what security indicators (if any) are used to warn users. Second, to examine how users react to spoofing emails and the impact of security indicators, we conduct a real-world phishing test in a user study. We have carefully applied “deception” to examine users’ natural reactions to the spoofing emails.

We also seek to understand why anti-spoofing protocols are not widely adopted, particularly from email providers’ perspectives. We planned to conduct a user study with email administrators from different institutions, which turned out to be challenging. Part of the reason is that the candidate pool is small. People who can provide insights for our questions need to have extensive experience managing real-world email services. In addition, email administrators often hesitate (or are not allowed) to share details about their anti-phishing/spoofing solutions. To these ends, we send our user study requests to 4000 email administrators of Alexa top domains. We eventually received responses from $N = 9$ administrators from various organizations (universities, payment services, online community websites) who agree to answer open questions either online or through in-person interviews.

1.2 Disposable Email Service

An Email address is one of the most important components of personally identifiable information (PII) on the Internet. Today’s online services typically require an email for account registration and password recovery. It’s easy to link an email address to the real-world identity. As a result, disposable email services have become a popular alternative which allows users to use online services without giving away their real email addresses. From disposable email services, a user can obtain a temporary email address without registration. After a short period of time, the emails will be disposed by the service providers. Users can use this disposable email address for certain tasks (e.g., registering an account on a dating website) without linking their online footprints to their real email addresses (e.g., work or personal email). In this way, potential attacks (e.g., spam, phishing, privacy leakage) will be drawn to the disposable addresses instead of the users’ real email accounts. Disposable email services are highly popular. For example, Guerrilla Mail, one of the earliest services, has processed 8 billion emails in the past decade [24].

While disposable email services allow users to hide their real identities, the email commu-
nication itself is not necessarily private. More specifically, most disposable email services maintain a public inbox, allowing any user to access any disposable email addresses at any time [30]. Essentially disposable email services are acting as a public email gateway to receive emails. The “public” nature not only raises interesting questions about the security of the disposable email service itself, but also presents a rare opportunity to empirically collect email data and study email tracking, a problem that is not well-understood.

In this thesis, we want to understand what disposable email services are used for in practice, and whether there are potential security or privacy risks involved with using a disposable email address.

1.3 Email Tracking

We use disposable email services as a public “honeypot” to collect emails sent by various online services and analyze email tracking in the wild. Unlike the extensively-studied web tracking [59, 60, 118, 194, 210, 250, 265], email tracking is not well-understood primarily due to a lack of large-scale email datasets. The largest study so far [120] has analyzed emails from 902 “Shopping” and “News” websites. In this thesis, we aim to significantly increase the measurement scale and uncover new tracking techniques.

1.4 Web Phishing

Phishing webpages, as the landing pages for phishing messages [115, 184, 248], are constantly involving to deceive users and evade detection. Sophisticated phishing pages are constructed to impersonate the webpages of banks, government agencies, and even the internal systems of major companies [103]. In addition, phishing pages can also impersonate the domain names of trusted entities via domain squatting techniques [147, 172, 217]. For example, an attacker may register a domain that looks like facebook.com using an internationalized domain name to deceive users, as shown in Figure 6.1. While anecdote evidence suggests such “elite” phishing pages exist, there is still a lack of in-depth understandings of how the phishing pages are constructed and used in practice.

In this thesis, we describe our efforts in searching and detecting squatting phishing domains where the attackers apply impersonation techniques to both the web content and the web domain. Our goals are threefold. First, we seek to develop a systematic method to search and detect squatting phishing domains in the wild. Second, we aim to empirically examine the impersonation techniques used by the attackers to deceive users. Third, we want to characterize the evasion techniques used by the squatting phishing pages and their effectiveness to avoid detection.

To these ends, we design a novel measurement system SquatPhito search and detect squat-
Chapter 1. Introduction

... ting phishing domains. We start by detecting a large number of “squatting” domains that are likely to impersonate popular brands. Then, we build a distributed crawler to collect the webpages and screenshots for the squatting domains. Finally, we build a machine learning classifier to identify squatting phishing pages. A key novelty is that our classifier is built based on a careful measurement of the evasion techniques used by real-world phishing pages. These evasion techniques are likely to render existing detection methods ineffective. Below, we describe each step and the discuss our key findings.

1.5 Credential Sharing

We perform an empirical measurement by piecing together the different stages of phishing to understand the information flow. We collect a large set of live phishing sites and feed fake login credentials to these sites. In this process, we monitor how the information is shared to the attackers who deployed the phishing site, and more importantly, any other third-parties. For the client-side measurement, we build a measurement tool to automatically detect a login form, fill in the fake credentials, and monitor the network traffic to external parties. For the phishing-server measurement, we build a crawler to retrieve phishing kits, and run them in a sandbox to detect first-party and third-party information collectors. Finally, to examine what attackers do after obtaining the login credentials, we set up our own honey accounts (in email services) to monitor the potential post-phishing exploiting activities. These steps allow us to provide an end-to-end view of the phishing process and credential sharing.

We performed the measurement from August 2018 to January 2019 covering 179,865 phishing URLs. The client-side measurement covers 41,986 live phishing sites, and the server-side measurement is based on the analysis of 2,064 detected phishing kits. Our post-phishing exploitation analysis uses 100 honey accounts from Gmail and 50 accounts from ProtonMail for data collection. We explore how likely attackers would attempt to use the leaked password to further hijack the associated email account.

1.6 Voice Personal Assistant

The ubiquitous usage of the Internet of Things (IoT) devices has proliferated the number of Voice Personal Assistant (VPA) systems in our home. As of Jan 2019, over 66.5 million households in the US [3] have one or more VPAs such as Amazon Alexa [67], Google Home [22], and Homepod [25]. The two dominating manufactures Amazon and Google introduce the voice assistant applications called “skills”. Third-party developers have built and published more than 84,000 skills worldwide in the application markets in 2019 [9, 20]. Users can “talk” to these applications to complete various tasks including opening a smart lock, starting their car, placing shopping orders, and transferring money to a friend. Although
these applications bring convenience, they also introduce new attack surfaces. Recent research shows that remote attackers can craft hidden voice commands to trigger the VPAs to launch malicious actions without user knowledge [254, 312, 316]. More recent work shows that attackers can publish malicious skills with similar pronunciations to fool the VPA to invoke the wrong application [182, 320]. Existing works have focused on proof-of-concept attacks by pointing out the potential ways of launching the attacks. However, there is a lack of empirical understanding of what functionality the third-party applications provide, and thus makes it difficult to systematically assess the consequences of these attacks.

In this thesis, we perform the first large-scale measurement on the third-party applications of Amazon Alexa and Google Home to systematically assess the attack surfaces. More specifically, given a voice assistant application, we seek to characterize its risk by detecting and analyzing the sensitive voice commands that are subject to potential attacks. Based on the recent proof-of-concept attacks [182, 254, 312, 316, 320], there are two main types of attack consequences: (1) controlling the system to perform an action, and (2) obtaining sensitive information. As such, we develop a natural language processing tool that classifies a given voice command from two dimensions. First, we examine whether a voice command is designed to insert an action (e.g., controlling a smart device) or retrieve information (e.g., obtaining user bank balance). Second, we classify whether the command is sensitive or nonsensitive. These two dimensions help to provide a more comprehensive view of the voice assistant skills, and their susceptibility to the existing attacks.

1.7 Contributions

Overall, this thesis uses data-driven methods to characterize and detect online deceptions. In the following, we highlight the specific research contributions.

Email Phishing. Our end-to-end measurement provides new insights into how email providers handle forged emails. We reveal the trade-offs between email availability and security made by different email providers. We are the first to empirically analyze the usage of security indicators on spoofed emails. We show that most email providers not only lack the necessary security indicators (particularly on mobile apps), but also have misleading UIs that help the attackers. We conduct a real-world phishing test to evaluate the effectiveness of the security indicator. We demonstrate the positive impact (and potential problems) of the security indicator and provide the initial guidelines for improvement.

In terms of anti-spoofing protocols adoption, we extracted and categorized 6 technical weaknesses in the existing anti-spoofing protocol designs based on our user study (and the protocol specifications). The result provides the taxonomy of the problem. Through the user study, we provide new insights into the perceived values and concerns of anti-spoofing protocols from email providers’ perspectives. These results shed light to the reasons behind the slow adoption of SPF, DKIM, and DMARC, pointing out the directions of improvement moving
forward. We discuss the key implication of the results to protocol designers, email providers, and users. We also discuss the possible solutions at the user-end to make up for the defective server-side authentication.

**Disposable Emails and Email Tracking** We perform the first measurement study on disposable email services by collecting a large-scale dataset (2.3 million emails) from 7 popular services over 3 months. Our analysis provides new insights into the common use cases of disposable email services and uncovers the potential risks of certain types of usage. We use the large-scale email dataset to empirically measure email tracking in the wild. We show the stealthy tracking methods used by third-party trackers collect data on user identifiers and user actions.

**Web Phishing** We propose a novel end-to-end measurement framework *SquatPhi* to search and detect squatting phishing pages from a large number of squatting domains. We perform the first in-depth analysis on squatting phishing domains in the wild. Our results provide insights into how squatting phishing pages impersonate popular brands at both the domain and content level. We empirically characterize the evasion techniques used by squatting phishing pages. The results indicate that existing detection methods are likely to be ineffective and need to be improved.

In terms of credentials sharing on phishing websites, we perform a large-scale empirical measurement on the information flow of credential sharing during phishing attacks. Our measurement covers both client-side, and server-side information sharing, and post-phishing exploitation. We build a new measurement tool to automatically seed fake credentials to phishing sites to measure the information sharing in real time. We will make the tool available for sharing with the research community. Our measurements provide new insights into the credential sharing mechanisms (to third-parties) during the phishing process.

**VPA** We perform the first large-scale empirical measurement on two dominating Voice Personal Assistant application markets, covering 82,770 skills and 211,843 voice commands. Our results provide new understandings of the capability and the sensitivity of the third-party applications. We identify a small set of sensitive applications (5.55%) that contributed to the vast majority of sensitive voice commands. We design and implement automated tools to classify VPA skills and their voice commands [28]. We perform a user survey with 400+ participants to measure the perceived sensitivity of voice commands.
Chapter 2

Background

2.1 SMTP and Email Spoofing

Today’s email system is built upon the SMTP protocol, which was initially designed without security in mind. Security extensions were introduced later to provide confidentiality, integrity, and authenticity. Below, we briefly introduce SMTP and related security extensions. Then we introduce our research questions and methodology.

Simple Mail Transfer Protocol (SMTP) is an Internet standard for electronic mail transmission [242]. Figure 2.1 shows the three main steps to deliver an email message. (1) Starting from the sender’s Mail User Agent (MUA), the message is first transmitted to the Mail Submission Agent (MSA) of the sender’s service provider via SMTP or HTTP/HTTPS. (2) Then the sender’s Mail Transfer Agent (MTA) sends the message to the receiver’s email provider using SMTP. (3) The message is then delivered to the receiving user by the Mail Delivery Agent (MDA) via Internet Message Access Protocol (IMAP), Post Office Protocol (POP) or HTTP/HTTPS.

When initially designed, SMTP did not have any security mechanisms to authenticate the sender identity. As a result, attackers can easily craft a forged email to impersonate/spoof an arbitrary sender address by modifying the “MAIL FROM” field in SMTP. Email spoofing is a critical step in a phishing attack — by impersonating a trusted entity as the email sender, the attacker has a higher chance to gain the victim’s trust. In practice, attackers usually exploit SMTP in step (2) by setting up their own MTA servers.

Alternatively, an attacker may also exploit step (1) if a legitimate email service is not carefully configured. For example, if a.com is configured as an open relay, attacker can use a.com’s server and IP to send forged emails that impersonate any email addresses.

2.1.1 Email Authentication

To defend against email spoofing attacks, various security extensions have been proposed and standardized including SPF, DKIM and DMARC. There are new protocols such as BIMI and ARC that are built on top of SPF, DKIM, and DMARC. In this chapter, we primarily focus on SPF, DKIM, and DMARC since they have some level of adoption by email services.
in practice. BIMI and ARC have not been fully standardized yet, and we will discuss them later in §3.7.

**SPF.**  Sender Policy Framework (SPF) allows an email service (or an organization) to publish a list of IPs that are authorized to send emails for its domain (RFC7208 [173]). For example, if a domain “a.com” published its SPF record in the DNS, then the receiving email services can check this record to match the sender IP with the sender email address. In this way, only authorized IPs can send emails as “a.com”. In addition, SPF allows the organization to specify a policy regarding how the receiver should handle the email that failed the authentication.

**DKIM.**  DomainKeys Identified Mail (DKIM) uses the public-key based approach to authenticate the email sender (RFC6376 [104]). The sender’s email service will place a digital signature in the email header signed by the private key associated to the sender’s domain. The receiving service can retrieve the sender’s public key from DNS to verify the signature. In order to query a DKIM public key from DNS, one not only needs the domain name but also a selector (an attribute in the DKIM signature). Selectors are used to permit multiple keys under the same domain for more a fine-grained signatory control. DKIM does not specify what actions that the receiver should take if the authentication fails.

**DMARC.**  Domain-based Message Authentication, Reporting and Conformance (DMARC) is built on top of SPF and DKIM (RFC7489 [204]), and it is not a standalone protocol. DMARC allows the domain administrative owner to publish a policy to specify what actions the receiver should take when the incoming email fails the SPF and DKIM check. In addition, DMARC enables more systematic reporting from receivers to senders. A domain’s DMARC record is available under _dmarc.domain.com in DNS.

### 2.1.2 The Low Adoption Rates of Anti-spoofing Protocols

In 2015, two measurement studies have shown that anti-spoofing protocols were not widely used among Internet domains [114, 131]. Among Alexa top 1 million domains [67], only 40% of the domains have published an SPF record and 1% have a DMARC record. DKIM is also
2.2 Disposable Email Services

Disposable email services are online web services where users can obtain a temporary email address to receive (or send) emails. After a short usage, the email address and its messages will be disposed by the service provider. Disposable email services allow users to register an online account without giving away their real email addresses. This helps to disconnect the user’s online activities from her real identity, and avoid attracting spam emails to the real email accounts.

There are two types of disposable email services, based on how temporal email addresses are assigned (Figure 2.2).

- **User-specified Addresses (UA).** Most services allow users to specify the username they want to use. For example, a user can obtain a temporary address “david@x.com” by specifying a username “david”. The user-specified address is more memorable for users.

- **Randomly-assigned Addresses (RA).** Some services create temporal email addresses for users by randomly generating usernames. For example, a user may be assigned to a random address that looks like “tt1hfd5m@x.com”. Users may refresh the web page to receive a different random address each time.

Figure 2.2: Two types of disposable email addresses.

not widely adopted based on Gmail’s internal estimation [114].

In January 2018, we conducted our own measurements to examine the recent adoption rates for SPF and DMARC, following the same methodology of [131], we find that among Alexa top 1 million domains, 44.9% of the domains have a valid SPF record and 5.1% of the domains have a valid DMARC record. Among the 1 million domains, 79% are Email domains with MX records. We find that 54.3% of the MX domains have a valid SPF record, and 6.0% of the MX domains have a valid DMARC record [153]. Compared with the study conducted in 2015, the adoption rates have increased, but only mildly. Our measurement result raises serious concerns about the effectiveness of the current spoofing defense. We are motivated to further explore the reasons behind the low adoption rates of anti-spoofing protocols.
While disposable email services allow users to temporarily use an email address, this email address and the received messages are not necessarily “private”. More specifically, most disposable email services are considered to be public email gateways, which means any users can see other users’ temporary inbox. For example, if a user A is using david@x.com at this moment, then another user B may also access the inbox of david@x.com at the same time. Very few disposable email services have implemented the sandbox mechanisms to isolate each temporary inbox. The only service we find that maintains a private inbox is inboxbear.com, which distinguishes each inbox based on the browser cookie. Therefore, many disposable email services have made it clear on their websites (or Terms of Services) that the email inbox is public and users should not expect privacy [29, 30].

2.3 Email Tracking

Email tracking is a method that allows the sender to know whether an email is opened by the receiver. A common method is to embed a small image (e.g., a 1×1 pixel) in the message body. When the receiver reads the email, the image will be automatically loaded by sending an HTTP or HTTPS request to a remote server. The remote server can be either the original email sender or a third-party service. In this way, the remote server will know when (based on timestamp) and where (based on IP) the email is read by which person (based on email address) using what device (based on “User-Agent”).

Email tracking is part of the broader category of web tracking. Web tracking, typically through third-party cookies and browser fingerprints, has been extensively studied [59, 60, 72, 87, 118, 128, 188, 194, 210, 223, 250, 262, 265]. However, very few studies have systematically examined email tracking because real-world email datasets are rarely available to researchers. The largest measurement study so far [120] collected data by signing up for “Shopping” and “News” websites to receive their emails. The resulting dataset contains 902 email senders. The limited number and category of online services severely limit researchers’ ability to draw generalizable conclusions.

We believe that the disposable email services provide a unique opportunity to study email tracking at a much larger scale and uncover new tracking techniques in the wild. First, disposable email services are public, which allows us to collect emails sent to disposable email addresses. Second, users of disposable email services have broadly exposed the email addresses to the Internet (by registering various online accounts), which helps to attract emails from a wide range of online services (and spammers). The resulting data, even though still has biases, is likely to be much more diversified.
2.4 Phishing Web Pages

We start by introducing the background of phishing, and defining elite phishing pages that apply squatting techniques.

Figure 2.3 shows the typical steps of a phishing attack. Attackers first need to trick users into visiting a phishing website. To gain the victim’s trust, a phishing website often impersonates other reputable services. In step 1, the victim user submits the login credential via the phishing page in the browser. After that, the information is then sent to the phishing server (step 2.1). The phishing server either directly sends the collected credentials via emails to the attacker (step 3.1), or the attacker will (manually) log into the phishing server to retrieve the information (step 3.2). Once the login credentials are obtained by the attacker, they can proceed further with malicious activities against users or their organizations (e.g., stealing data, compromising enterprise/government networks).

Phishing has been widely used by cybercriminals to steal user credentials and breach large networks. Typically, attackers would impersonate a trusted entity to gain the victim’s trust, luring the victim to reveal important information. Phishing pages often act as the landing pages of malicious URLs distributed by phishing emails [184], SMS [248], or social network messages [115]. The phishing pages usually contain a form to trick users to enter passwords or credit card information.

As phishing attacks become prevalent [32], various phishing detection methods have been proposed, ranging from URL blacklisting [79] to visual similarity based phishing detection [213] and website content-based classification [303]. Visual similarity-based phishing detection [213] aims to compare the original webpages of popular brands to suspicious pages to detect “impersonation”. Machine learning based methods [303] rely on features extracted from the HTML source code, JavaScript, and the web URLs to flag phishing websites. As
phishing attacks evolve, we are curious about the potential evasion techniques used by attackers in practice.

**Domain Name Squatting.** Domain name squatting is the act of registering domain names that are likely to cause confusions with existing brands and trademarks. Domain name squatting has led to abusive activities such as impersonating the original websites to steal traffic, and distribute ads and malware. A squatting domain usually shares many overlapping characters at a targeted domain. Common squatting techniques include bit mutation [224], typo spelling [217] and homograph imitating [172]. Internationalized domain names (IDN) can be used for domain squatting domains, since IDNs can have a similar visual representation as the target domains after encoding (Figure 6.1).

Squatting domains can cause trouble to users as well as the target brands. For example, users often mis-type the domain name of the website they want to visit in the address bar (e.g., typing facebook.com for facebook.com). As a result, users could be visiting a website hosted under a squatting domain. Speculators register squatting domains of popular brands and resell them at a much higher price. Sometimes, popular brands (e.g., big banks) have to purchase squatting domains that targeting their websites so that they can redirect users back to the correct websites [41].

**Domain Squatting for Phishing.** Squatting domains are naturally powerful to conduct phishing attacks since the domain name looks similar to that of a trusted website. We refer phishing pages hosted under squatting domains as squatting phishing pages. More formally, a squatting phishing page ($P_s$) has two properties: (1) it has a squatting-based domain ($S$); and (2) its webpage contains deceptive phishing content ($W$). $P_s = S \lor W$.

**Phishing Kits.** Attackers often deploy phishing websites using a collection of software tools called phishing kits [100]. Phishing kits allow people with little technical skills to run phishing attacks. A typical kit contains a website component, and an information processing component. The website component contains the code, images, and other content to create a fake website. The information processing tool will automatically record and store the received information (password, login time, IP), and send the information to the attacker. Some phishing kits also contain a spamming tool, which can send spam emails to lead users to the phishing sites.

**Third-party Information Sharing.** During a phishing attack, it is possible that the user credentials are also shared to third-parties, in both the *client-side* and the *server-side*.

- **Client-side Third Parties.** Step2.2 shows that client-side third-parties collect the user credential. In this case, the phishing server that directly hosts the phishing page is the first-party and any other servers that also collect the credential are third-parties. The information sharing happens in real time when the user clicks on the “submit” button.
2.5 Voice Personal Assistant

Voice Personal Assistant (VPA) is a software agent, which provides aids for individuals, like setting calendar events, making reservations, controlling smart home devices. Most VPAs such as Amazon Alexa and Google Home use a cloud-based model to host skills and interact with users. The workflow is shown in Figure 2.4. After the user talks to the VPA device (1), the voice command is first sent to the Amazon/Google Cloud (2). The cloud needs to translate the natural language command into an API call, and then sends the API call to the corresponding skill servers. To develop a skill, third-party developers can either host their skill servers directly within the cloud, or they can run the server independently. Regardless of which way the developers choose, they typically rely on the cloud to parse and interpret the voice command and route the API calls. The natural language process system in the cloud is the main target (or weak link) to launch attacks. We will introduce specific threat models in the next section. Some of the skills are used to control other smart home devices. In this case, either the Amazon/Google cloud or the skill server will send the request to the smart devices to perform tasks or configure the settings (3–4). After processing, the operation status or response will be sent back to users (5–6).

Some of the skills require the user to create an account (e.g., shopping services, banks, smart device services). Both Amazon Alexa and Google Home use OAuth to link the user’s
skill-specific account to the user’s voice assistant device so that users can interact with the
skill service through the VPA system. This mechanism is called account linking. Many
simple skills such as games, quizzes, and question-answering typically do not require account
linking.

Google Home and Amazon Alexa maintain their “app stores” where each application (skill)
has its own web page. For each skill, the web page shows the voice commands associated
with that particular skill. Note that Amazon Alexa only allows up to three voice commands
listed as example commands (five commands for Google Home skills). As such, developers
would list the most important commands in the application descriptions in the introduction
page.
3.1 Introduction

Despite the recent development of the system and network security, human factors still remain a weak link. As a result, attackers increasingly rely on phishing tactics to breach various target networks [275] and cause large data breaches [94, 260, 261]. For example, email phishing has involved in nearly half of the 2000+ reported security breaches in recent two years, causing a leakage of billions of user records [288].

Email spoofing is a critical step in phishing, where the attacker impersonates a trusted entity to gain the victim’s trust. According to the recent report from the Anti-Phishing Working Group (APWG), email spoofing is widely in spear phishing attacks to target employees of various businesses [12]. Unfortunately, today’s email transmission protocol (SMTP) has no built-in mechanism to prevent spoofing [242]. It relies on email providers to implement SMTP extensions such as SPF [173], DKIM [104] and DMARC [204] to authenticate the sender. Since implementing these extensions is voluntary, their adoption rate is far from satisfying. Real-world measurements conducted in 2015 have shown that among Alexa top 1 million domains, 40% have SPF, 1% have DMARC, and even fewer are correctly/strictly configured [114, 131].

The limited server-side protection is likely to put users in a vulnerable position. Since not every sender domain has adopted SPF/DKIM/DMARC, email providers still face key challenges to reliably authenticate all the incoming emails. When an email failed the authentication, it is a “blackbox” process in terms of how email providers handle this email. Would forged emails still be delivered to users? If so, how could users know the email is questionable? Take Gmail for example, Gmail delivers certain forged emails to the inbox and places a security indicator on the sender icon (a red question mark, Figure 3.5(a)). We are curious about how a broader range of email providers handle forged emails, and how much the security indicators actually help to protect users.

In this chapter, we describe our efforts and experience in evaluating the real-world defenses against email spoofing\(^1\). We answer the above questions through empirical end-to-end spoofing measurements, and a user study. First, we conduct measurements on how popular email providers detect and handle forged emails. The key idea is to treat each email provider as

---

\(^1\)Our study has been approved by our local IRB (IRB-17-397).
Chapter 3. Email Spoofing Attacks

a blackbox and vary the input (forged emails) to monitor the output (the receiver’s inbox). Our goal is to understand under what conditions the forged/phishing emails are able to reach the user inbox and what security indicators (if any) are used to warn users. Second, to examine how users react to spoofing emails and the impact of security indicators, we conduct a real-world phishing test in a user study. We have carefully applied “deception” to examine users’ natural reactions to the spoofing emails.

**Measurements.** We start by scanning Alexa top 1 million hosts from February 2017 to January 2018. We confirm that the overall adoption rates of SMTP security extensions are still low (SPF 44.9%, DMARC 5.1%). This motivates us to examine how email providers handle incoming emails that failed the authentication.

We conduct end-to-end spoofing experiments on 35 popular email providers used by billions of users. We find that forged emails can penetrate the majority of email providers (34/35) including Gmail, Yahoo Mail and Apple iCloud under proper conditions. Even if the receiver performs all the authentication checks (SPF, DKIM, DMARC), spoofing an unprotected domain or a domain with “relaxed” DMARC policies can help the forged email to reach the inbox. In addition, spoofing an “existing contact” of the victim also helps the attacker to penetrate email providers (e.g., Hotmail).

More surprisingly, while most providers allow forged emails to get in, rarely do they warn users of the unverified sender. Only 9 of 35 providers have implemented some security indicators: 8 providers have security indicators on their web interface (e.g., Gmail) and only 4 providers (e.g., Naver) have the security indicators consistently for the mobile apps. There is no security warning if a user uses a third-party email client such as Microsoft Outlook. Even worse, certain email providers have misleading UI elements which help the attacker to make forged emails look authentic. For example, when attackers spoof an existing contact (or a user from the same provider), 25 out of 35 providers will automatically load the spoofed sender’s photo, a name card or the email history along with the forged email. These UI designs are supposed to improve the email usability, but in turn, help the attacker to carry out the deception when the sender address is actually spoofed.

**Phishing Experiment.** While a handful of email providers have implemented security indicators, the real question is how effective they are. We answer this question using a user study (N = 488) where participants examine spoofed phishing emails with or without security indicators on the interface. This is a real-world phishing test where deception is carefully applied such that users examine the spoofed emails without knowing that the email is part of an experiment (with IRB approval). We debrief the users and obtain their consent after the experiment.

Our result shows that security indicators have a positive impact on reducing risky user actions but cannot eliminate the risk. When a security indicator is not presented (the controlled group), out of all the users that opened the spoofed email, 48.9% of them eventually clicked on the phishing URL in the email. For the other group of users to whom we present the
security indicator, the corresponding click-through rate is slightly lower (37.2%). The impact is consistently positive for users of different demographics (age, gender, education level). On the other hand, given the 37.2% click-through rate, we argue that the security indicator cannot eliminate the phishing risk. The server-side security protocols and the user-end security indicators should be both improved to maximize the impact.

Contributions. We have 3 key contributions:

- **First**, our end-to-end measurement provides new insights into how email providers handle forged emails. We reveal the trade-offs between email availability and security made by different email providers.
- **Second**, we are the first to empirically analyze the usage of security indicators on spoofed emails. We show that most email providers not only lack the necessary security indicators (particularly on mobile apps), but also have misleading UIs that help the attackers.
- **Third**, we conduct a real-world phishing test to evaluate the effectiveness of the security indicator. We demonstrate the positive impact (and potential problems) of the security indicator and provide the initial guidelines for improvement.

The quantitative result in this chapter provides an end-to-end view on how spoofed emails could penetrate major email providers and all the way affect the end users. We hope the results can draw more attention from the community to promoting the adoption of SMTP security extensions. In addition, we also seek to raise the attention of email providers to designing and deploying more effective UI security indicators, particularly for the less protected mobile email apps. We have communicated the results with the Gmail team and offered suggestions to improve the security indicators.

### 3.2 Research Questions and Methodology

Despite the available security mechanisms, significant challenges remain when these mechanisms are not properly deployed in practice. Measurements conducted in 2015 show that the adoption rates of SMTP security extensions are far from satisfying [114, 131]. Among Alexa top 1 million domains, only 40% have published an SPF record, and only 1% have a DMARC policy. These results indicate a real challenge to protect users from email spoofing. First, with a large number of domains not publishing an SPF/DKIM record, email providers cannot reliably detect incoming emails that spoof unprotected domains. Second, even a domain is SPF/DKIM-protected, the lack of (strict) DMARC policies puts the receiving server in a difficult position. It is not clear how the email providers at the receiving end would handle unverified emails. Existing works [114, 131] mainly focus on the authentication protocols on the server-side. However, there is still a big gap between the server-side detection and the actual impact on users.
Our Questions. Our study seeks to revisit the email spoofing problem by answering three key questions. (1) When email providers face uncertainty in authenticating incoming emails, how would they handle the situation? Under what conditions would forged emails be delivered to the users? (2) Once forged emails reach the inbox, what types of warning mechanisms (if any) are used to notify users of the unverified sender address? (3) How effective is the warning mechanism? Answering these questions is critical to understanding the actual risks exposed to users by spoofing attacks.

We answer question(1)–(2) through end-to-end spoofing experiments (§3.3, §3.4 and §3.5). For a given email provider, we treat it as a “blackbox”. By controlling the input (e.g., forged emails) and monitoring the output (receiver’s inbox), we infer the decision-making process inside the blackbox. We answer question(3) by conducting a large user study (§3.6). The idea is to let users read spoofing/phishing emails with and without security indicators.

Ethics. We have taken active steps to ensure research ethics. Our measurement study only uses dedicated email accounts owned by the authors and there is no real user getting involved. In addition, to minimize the impact on the target email services, we have carefully controlled the message sending rate (one message every 10 minutes), which is no different than a regular email user. For the user study that involves “deception”, we worked closely with IRB for the experiment design. More detailed ethical discussions are presented later.

<table>
<thead>
<tr>
<th>Status</th>
<th>All Domain # (%)</th>
<th>MX Domain # (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total domains</td>
<td>1,000,000 (100%)</td>
<td>792,556 (100%)</td>
</tr>
<tr>
<td>w/ SPF</td>
<td>492,300 (49.2%)</td>
<td>473,457 (59.7%)</td>
</tr>
<tr>
<td>w/ valid SPF</td>
<td><strong>448,741 (44.9%)</strong></td>
<td><strong>430,504 (54.3%)</strong></td>
</tr>
<tr>
<td>Policy: soft fail</td>
<td>272,642 (27.3%)</td>
<td>268,317 (33.9%)</td>
</tr>
<tr>
<td>Policy: hard fail</td>
<td><strong>125,245 (12.5%)</strong></td>
<td><strong>112,415 (14.2%)</strong></td>
</tr>
<tr>
<td>Policy: neutral</td>
<td>49,798 (5.0%)</td>
<td>48,736 (6.1%)</td>
</tr>
<tr>
<td>Policy: pass</td>
<td>1,056 (0.1%)</td>
<td>1,036 (0.1%)</td>
</tr>
<tr>
<td>w/ DMARC</td>
<td>51,222 (5.1%)</td>
<td>47,737 (6.0%)</td>
</tr>
<tr>
<td>w/ valid DMARC</td>
<td><strong>50,619 (5.1%)</strong></td>
<td><strong>47,159 (6.0%)</strong></td>
</tr>
<tr>
<td>Policy: none</td>
<td>39,559 (4.0%)</td>
<td>36,984 (4.7%)</td>
</tr>
<tr>
<td>Policy: reject</td>
<td><strong>6,016 (0.6%)</strong></td>
<td><strong>5,225 (0.7%)</strong></td>
</tr>
<tr>
<td>Policy: quarantine</td>
<td>5,044 (0.5%)</td>
<td>4,950 (0.6%)</td>
</tr>
</tbody>
</table>

Table 3.1: SPF/DMARC statistics of Alexa 1 million domains. The data was collected in January 2018.
3.3. Adoption of SMTP Extensions

The high-level goal of our measurement is to provide an end-to-end view of email spoofing attacks against popular email providers. Before doing so, we first examine the recent adoption rate of SMTP security extensions compared with that of three years ago [114, 131]. This helps to provide the context for the challenges that email providers face to authenticate incoming emails.

Scanning Alexa Top 1 Million Domains. Email authentication requires the sender domains to publish their SPF/DKIM/DMARC records to DNS. To examine the recent adoption rate of SPF and DMARC, we crawled 3 snapshots the DNS record for Alexa top 1 million hosts [67] in February 2017, October 2017, and January 2018. Similar to [114, 131], this measurement cannot apply to DKIM, because querying the DKIM record requires knowing the selector information for every each domain. The selector information is only available in the DKIM signature in the email header, which is not a public information. We will measure the DKIM usage later in the end-to-end measurement.

Recent Adoption Rates. Table 3.1 shows the statistics for the most recent January 2018 snapshot. SPF and DMARC both have some increase in the adoption rate but not very significant. About 44.9% of the domains have published a valid SPF record in 2018 (40% in 2015 [131]), and 5.1% have a valid DMARC record in 2018 (1.1% in 2015 [131]). The invalid records are often caused by the domain administrators using the wrong format for the SPF/DMARC record. Another common error is to have multiple records for SPF (or DMARC), which is equivalent to “no record” according to RFC7489 [204]. Figure 3.1 shows the adoption rate for all three snapshots. Again, the adoption rates have been increasing at a slow speed.

Among the 1 million domains, 792,556 domains are MX domains (i.e., mail exchanger do-
mains that host email services). The adoption rates among MX domains are slightly higher (SPF 54.3%, DMARC 6.0%). For non-MX domains, we argue that it is also important to publish the SPF/DMARC record. For example, office.com is not a MX domain, but it hosts the website of Microsoft Office. Attackers can spoof office.com to phish Microsoft Office users or even the employees.

**Failing Policy.** SPF and DMARC both specify a policy regarding what actions the receiver should take after the authentication fails. Table 3.1 shows that only a small portion of the domains specifies a strict “reject” policy: 12.5% of the domains set “hard fail” for SPF, and 0.6% set “reject” for DMARC. The rest of the domains simply leave the decision to the email receiver. “Soft fail”/“quarantine” means that the email receiver should process the email with caution. “Neutral”/“none” means that no policy is specified. SPF’s “pass” means that the receiver should let the email go through. If a domain has both SPF and DMARC policies, DMARC overwrites SPF as long as the DMARC policy is not “none”.

Domains that use DKIM also need to publish their policies through DMARC. The fact that only 5.1% of the domains have a valid DMARC record and 0.6% have a “reject” policy indicates that most DKIM adopters also did not specify a strict reject policy.

**Popular Domains.** Not too surprisingly, popular domains’ adoption rates are higher as shown in Figure 3.2. We divide the top 1 million domains into log-scale sized bins. For SPF, the top 1,000 domains have an adoption rate of 73%. For DMARC, the adoption rate of top 1000 domains is 41%. This indicates that administrators of popular domains are more motivated to prevent their domains from being spoofed. Nevertheless, there is still a large number of (popular) domains remain unprotected.

![Figure 3.2: The adoption rate as a function of the domains’ Alexa rankings (January 2018).](image)
3.4 End-to-End Spoofing Experiments

Given the current adoption rate of SMTP extension protocols, it is still challenging for email providers to reliably authenticate all incoming emails. When encountering questionable emails, we are curious about how email providers make such decisions. In the following, we describe the details of our measurement methodology and procedures.

3.4.1 Experiment Setup

We conduct end-to-end spoofing experiments on popular email providers that are used by billions of users. As shown in Figure 3.3, for a given email provider (B.com), we set up a user account under B.com as the email receiver (test@B.com). Then we set up an experimental server (E.com) to send forged emails to the receiver account. Our server runs a Postfix mail service [1] to directly interact with the target mail server using SMTP. By controlling the input (the forged email) and observing the output (the receiver account), we infer the decision-making process inside of the target email service.

Selecting Target Email Providers. This study focuses on popular and public email services with two considerations. First, popular email services such as Yahoo Mail and Gmail are used by more than one billion users [189, 238]. Their security policies and design choices are likely to impact more people. Second, to perform end-to-end experiments, we need to collect data from the receiver end. Public email services allow us to create an account as the receiver. Our experiment methodology is applicable to private email services but requires collaborations from the internal users.

To obtain a list of popular public email services, we refer to Adobe’s leaked user database (152 million email addresses, 9.3 million unique email domains) [176]. We ranked the email domains based on popularity, and manually examined the top 200 domains (counting for 77.7% of all email addresses). After merging domains from the same service (e.g., hotmail.com and outlook.com) and excluding services that don’t allow us to create an account, we obtained a short list of 28 email domains. To include the more recent public email services, we searched on Google and added 6 more services (yeah.net, protonmail.com, tutanota.com, zoho.com, fastmail.com, and runbox.com). We notice that Google’s Gmail and Inbox have very different email interfaces and we treat them as two services.

In total, we have 35 popular email services which cover 99.8 million email addresses (65.7%) in the Adobe database. As an additional reference, we also analyze the Myspace database (131.4 million email addresses) [237]. We find that 101.8 million email addresses (77.5%) are from the 35 email services, confirming their popularity. The list of the email providers is shown in Table 3.2.
Chapter 3. Email Spoofing Attacks

Figure 3.3: End-to-end spoofing experiment setup. We use our server E.com to send a forged email to the target email service B.com by spoofing A.com.

3.4.2 Experiment Parameters

To examine how different factors affect the outcome of email spoofing, we apply different configurations to the experiment. We primarily focus on parameters that are likely to affect the spoofing outcome, including the spoofed sender address, email content, sender IP, and the receiver’s email client (user interface).

Spoofed Sender Address. The sender address is a critical part of the authentication. For example, if the spoofed domain (A.com) has a valid SPF/DKIM/DMARC record, then the receiver (in theory) is able to detect spoofing. We configure three profiles for the spoofed sender domain: (1) None: no SPF/DKIM/DMARC record (e.g., thepiratebay.org); (2) Relaxed: SPF/DKIM with a “none” policy (e.g., tumblr.com); and (3) Strict: SPF/DKIM with a strict “reject” policy (e.g., facebook.com). For each profile, we randomly pick 10 domains (30 domains in total) from Alexa top 5000 domains (the detailed list is in Appendix A).

Email Content. Email content can affect how spam filters handle the incoming emails [77]. Note that our experiment is not to reverse-engineer exactly how spam filters weight different keywords, which is an almost infinite searching space. Instead, we focus on spoofing (where the sender address is forged). We want to minimize the impact of spam filters and examine how the receivers’ decision is affected by the address forgery (spoofing) alone.

To this end, we configure 5 different types of email content for our study: (1) a blank email, (2) a blank email with a benign URL (http://google.com), (3) a blank email with a benign attachment (an empty text file). Then we have (4) a benign email with actual content. This email is a real-world legitimate email that informs a colleague about the change of time for a meeting. The reason for using “benign” content is to test how much the “spoofing” factor alone contributes to the email providers’ decisions. In addition, to test whether a phishing email can penetrate the target service, we also include (5) an email with phishing content. This phishing email is a real-world sample from a phishing attack targeting our institution recently. The email impersonates the technical support to notify the victim that her internal
3.5. Spoofing Experiment Results

account has been suspended and ask her to re-activate the account using a URL (to an Amazon EC2 server).

Sender IP. The IP address of the sender’s mail server may also affect the spoofing success. We configure a static IP address and a dynamic IP address. Typically, mail servers need to be hosted on a static IP. In practice, attackers may use dynamic IPs for the lower cost.

Email Client. We examine how different email clients warn users of forged emails. We consider 3 common email clients: (1) a web client, (2) a mobile app, and (3) a third-party email client. All the 35 selected services have a web interface, and 28 have a dedicated mobile app. Third-party clients refer to the email applications (e.g., Microsoft Outlook and Apple Mail) that allow users to check emails from any email providers.

3.5 Spoofing Experiment Results

In this section, we describe the results of our experiments. First, to provide the context, we measure the authentication protocols that the target email providers use to detect forged emails. Then, we examine how email providers handle forged emails and identify the key factors in the decision making. For emails that reached the inbox, we examine whether and how email providers warn users about their potential risks. Note that in this section, the all experiment results reflect the state of the target email services as of January 2018.

3.5.1 Authentication Mechanisms

To better interpret the results, we first examine how the 35 email providers authenticate incoming emails. One way of knowing their authentication protocols is to analyze the email headers and look for SPF/DKIM/DMARC authentication results. However, not all the email providers add the authentication results to the header (e.g., qq.com) Instead, we follow a more reliable method [131] by setting up an authoritative DNS server for our own domain and sending an email from our domain. In the meantime, the authoritative DNS server will wait and see whether the target email service will query the SPF/DKIM/DMARC record. We set the TTL of the SPF, DKIM and DMARC records as 1 (second) to force the target email service always querying our authoritative DNS server. The results are shown in Table 3.2 (left 4 columns). 35 email providers can be grouped into 3 categories based on their protocols:

- **Full Authentication (16)**: Email services that perform all three authentication checks (SPF, DKIM and DMARC). This category includes the most popular email services such as Gmail, Hotmail and iCloud.

- **SPF/DKIM but no DMARC (15)**: Email services that check either SPF/DKIM, but do not check the sender’s DMARC policy. These email services are likely to make
<table>
<thead>
<tr>
<th>Email Provider</th>
<th>Protocol</th>
<th>Overall Rate n=1500</th>
<th>IP ST</th>
<th>IP DYN</th>
<th>Spoofed Address Profile None Relaxed Strict</th>
<th>BLK</th>
<th>Email Content URL</th>
<th>Atta.</th>
<th>Ben.</th>
<th>Phi.</th>
</tr>
</thead>
<tbody>
<tr>
<td>mail.ru</td>
<td>✓ ✓ ✓</td>
<td>0.69</td>
<td>0.69</td>
<td>0.69</td>
<td>1.00 0.99 0.07</td>
<td>0.70</td>
<td>0.69 0.68 0.68</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fastmail.com</td>
<td>✓ ✓ ✓</td>
<td>0.66</td>
<td>1.00</td>
<td>0.32</td>
<td>0.70 0.65 0.64</td>
<td>0.67</td>
<td>0.66 0.67 0.65</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>163.com</td>
<td>✓ ✓ ✓</td>
<td>0.58</td>
<td>0.66</td>
<td>0.50</td>
<td>0.73 0.54 0.47</td>
<td>0.53</td>
<td>0.60 0.46 0.66</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>126.com</td>
<td>✓ ✓ ✓</td>
<td>0.57</td>
<td>0.66</td>
<td>0.48</td>
<td>0.74 0.54 0.43</td>
<td>0.54</td>
<td>0.56 0.46 0.65</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>gmail.com</td>
<td>✓ ✓ ✓</td>
<td>0.53</td>
<td>0.56</td>
<td>0.51</td>
<td>0.93 0.66 0.00</td>
<td>0.58</td>
<td>0.58 0.50 0.60</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>gmail inbox</td>
<td>✓ ✓ ✓</td>
<td>0.53</td>
<td>0.56</td>
<td>0.51</td>
<td>0.93 0.66 0.00</td>
<td>0.58</td>
<td>0.58 0.50 0.60</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>naver.com</td>
<td>✓ ✓ ✓</td>
<td>0.50</td>
<td>0.50</td>
<td>0.51</td>
<td>0.95 0.56 0.00</td>
<td>0.51</td>
<td>0.50 0.50 0.50</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>yeah.net</td>
<td>✓ ✓ ✓</td>
<td>0.36</td>
<td>0.51</td>
<td>0.21</td>
<td>0.44 0.38 0.26</td>
<td>0.23</td>
<td>0.35 0.34 0.61</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tutanota.com</td>
<td>✓ ✓ ✓</td>
<td>0.36</td>
<td>0.41</td>
<td>0.30</td>
<td>0.90 0.17 0.00</td>
<td>0.39</td>
<td>0.39 0.20 0.39</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>yahoo.com</td>
<td>✓ ✓ ✓</td>
<td>0.35</td>
<td>0.67</td>
<td>0.03</td>
<td>0.52 0.52 0.00</td>
<td>0.33</td>
<td>0.34 0.33 0.38</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>inbox.lv</td>
<td>✓ ✓ ✓</td>
<td>0.32</td>
<td>0.63</td>
<td>0.00</td>
<td>0.50 0.45 0.00</td>
<td>0.32</td>
<td>0.32 0.32 0.32</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>protonmail.com</td>
<td>✓ ✓ ✓</td>
<td>0.30</td>
<td>0.60</td>
<td>0.00</td>
<td>0.45 0.45 0.00</td>
<td>0.32</td>
<td>0.26 0.29 0.31</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>seznam.cz</td>
<td>✓ ✓ ✓</td>
<td>0.24</td>
<td>0.48</td>
<td>0.00</td>
<td>0.35 0.25 0.13</td>
<td>0.35</td>
<td>0.35 0.35 0.08</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aol.com</td>
<td>✓ ✓ ✓</td>
<td>0.18</td>
<td>0.16</td>
<td>0.19</td>
<td>0.29 0.25 0.00</td>
<td>0.24</td>
<td>0.20 0.22 0.23</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>icloud.com</td>
<td>✓ ✓ ✓</td>
<td>0.07</td>
<td>0.10</td>
<td>0.04</td>
<td>0.11 0.09 0.00</td>
<td>0.01</td>
<td>0.01 0.01 0.17</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>hotmail.com</td>
<td>✓ ✓ ✓</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00 0.00 0.00</td>
<td>0.00</td>
<td>0.00 0.00 0.00</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3.2: The ratio of emails that reached the inbox (inbox rate). We break down the inbox rate for emails with different configuration parameters (sender IP, the SPF/DKIM/DMARC profile of the sender address, and the email content).

decisions on their own.

- **No Authentication (4):** Email services that do not perform any of the three authentication protocols.

### 3.5.2 Decisions on Forged Emails

Next, we examine the decision-making process on forged emails. For each of the 35 target email services, we test all the possible combinations of the parameter settings (30 spoofed addresses × 5 types of email content × 2 IP addresses), and then repeat the experiments for 5 times. Each email service receives 300 × 5 = 1,500 emails (52,500 emails in total).
3.5. Spoofing Experiment Results

Figure 3.4: The aggregated ratio of emails that reached the user inbox (inbox rate). The legend displays the 3 authentication groups of the receivers. Each subfigure shows the breakdown results for emails with specific configurations.

We shuffled all the emails and send them in randomized orders. We also set a sending time interval of 10 minutes (per email service) to minimize the impact to the target mail server. The experiment was conducted in December 2017–January 2018. Note the volume of emails in the experiment is considered very low compared to the hundreds of billions of emails sent over the Internet every day [50]. We intentionally limit our experiment scale so that the experiment emails would not impact the target services (and their email filters) in any significant ways. The randomized order and the slow sending speed helps to reduce the impact of the earlier emails to the later ones in the experiments.

After the experiment, we rely on IMAP/POP to retrieve the emails from the target email provider. For a few providers that do not support IMAP or POP, we use a browser-based crawler to retrieve the emails directly through the web interface. As shown in Table 3.2, we group email providers based on the supported authentication protocols. Within each group, we rank email providers based on the inbox rate, which is the ratio of emails that arrived the inbox over the total number of emails sent. Emails that did not arrive the inbox were either placed in the spam folder or completely blocked by the email providers.

Ratio of Emails in the Inbox. Table 3.2 shows that the vast majority of email services can be successfully penetrated. 34 out of the 35 email services allowed at least one forged email to arrive the inbox. The only exception is Hotmail which blocked all the forged emails. 33 out of 35 services allowed at least one phishing email to get into the inbox. In particular, the phishing email has penetrated email providers that perform full authentications (e.g., Gmail, iCloud, Yahoo Mail) when spoofing sender domains that do not have a strict reject DMARC policy. In addition, providers such as juno.com, t-online.de, and excite.com did not block forged emails at all with a 100% inbox rate. juno.com actually checked both SPF and DKIM. This suggests that even though the email providers might have detected the email forgery, they still deliver the email to the user inbox.
Impact of Receiver’s Authentication. Table 3.2 shows that email providers’ authentication methods affect the spoofing result. For email providers that perform no authentication, the aggregated inbox rate is 94.2%. In comparison, the aggregated inbox rate is much lower for email providers that perform a full authentication (39.0%) and email providers that just perform SPF/DKIM (39.3%). To examine the statistical significance of the differences, we apply Chi-Squared test on emails sent to the three types of email providers. The result confirms that emails are more likely to reach the inbox of “no-authentication” providers compared to the two other groups with statistical significance (both $p < 0.01$).

However, the difference between the “full-authentication” email providers and the “SPF/DKIM only” email providers are not statistically significant ($p = 0.495$). This indicates that the DMARC check has a relatively minor effect. Table 3.2 shows that DMARC check primarily affects emails where the spoofed domain has a “strict” reject policy. However, even with a full-authentication, the inbox rate of these emails is not always 0.00 (e.g., mail.ru, fastmail.com, 163.com, 126.com, yeah.net, seznam.cz). This is because certain email providers would consider the DMARC policy as a “suggested action”, but do not always enforce the policy.

Impact of the Sender IP. To better illustrate the impact of different email configurations, we plot Figure 3.4. We first group the target email providers based on their authentication method (3 groups), and then calculate the aggregated inbox rate for a specific configuration setting. As shown in Figure 3.4a, emails that sent from a static IP has a higher chance to reach the inbox (56.9%) compared to those from a dynamic IP (33.9%). Chi-Square statistical analysis shows the difference is statistically significant ($p < 0.0001$). In practice, however, dynamic IPs are still a viable option for attackers since they are cheaper.

To ensure the validity of results, we have performed additional analysis to make sure our IPs were not blacklisted during the experiment. More specifically, we analyze our experiment traces to monitor the inbox rate throughout the experiment process. In our experiment, each email service receives 1500 emails, and we checked the inbox rate per 100 emails over time. If our IPs were blacklisted during the experiment, there should be a sharp decrease in the inbox rate at some point. We did not observe that in any of the tested email services. We also checked 94 public blacklists, and our IPs are not on any of them.

Impact of Spoofed Sender Domain. Figure 3.4b demonstrates the impact of the spoofed sender address. Overall, spoofing a sender domain that has no SPF/DKIM/DMARC records yields a higher inbox rate (60.5%). Spoofing a sender domain with SPF/DKIM and a “relaxed” failing policy has a lower inbox rate (47.3%). Not too surprisingly, domains with SPF/DKIM records and a “strict” reject policy is the most difficult to spoof (inbox rate of 28.4%). Chi-Square statistical analysis shows the differences are significant ($p < 0.00001$). The result confirms the benefits of publishing SPF/DKIM/DMARC records. However, publishing these records cannot completely prevent being spoofed, since email providers may

---

2https://mxtoolbox.com/blacklists.aspx
3.5. Spoofing Experiment Results

<table>
<thead>
<tr>
<th>Feature</th>
<th>$\chi^2$</th>
<th>Mutual Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receiver authentication method</td>
<td>6497.93</td>
<td>0.0707</td>
</tr>
<tr>
<td>Spoofed sender address</td>
<td>3658.72</td>
<td>0.0356</td>
</tr>
<tr>
<td>Sender IP</td>
<td>2799.51</td>
<td>0.0269</td>
</tr>
<tr>
<td>Email content</td>
<td>115.27</td>
<td>0.0011</td>
</tr>
</tbody>
</table>

Table 3.3: Feature ranking.

Figure 3.5: Security indicators on forged emails from 9 email providers. (a)–(e) are for regular forged emails. (f)–(h) only show up when the spoofed sender and the receiver belong to the same provider. (i) only shows up when spoofing an existing contact.

still deliver emails that failed the SPF/DKIM authentication.

Impact of Email Content. Figure 3.4c shows that the inbox rates are not very different for different email content. The differences are small but not by chance ($\chi^2$-Squared test $p < 0.00001$). This suggests that our result is not dependent on a specific email content chosen for the study. Recall that we specifically use benign-looking content to minimize the impact of spam filters, so that we can test how much the “spoofing” factor contributes to email providers’ decisions. This does not mean that email content has no impact on the decision making. On the contrary, if an email has a blacklisted URL or a known malware as the attachment, we expected more emails will be blocked (which is not our study purpose). Our result simply shows that today’s attackers can easily apply spoofing to conduct targeted spear phishing. In the context of spear phishing, it is a reasonable assumption that the attacker will craft benign-looking content with URLs that have not been blacklisted yet [146].

Ranking the Factors. To determine which factors contribute more to a successful penetration, we perform a “feature ranking” analysis. We divide all the emails into two classes: positive (inbox) and negative (spam folder or blocked). For each email, we calculate four features: email content ($F_1$), sender address profile ($F_2$), receiver authentication group ($F_3$), and sender IP ($F_4$), all of which are categorical variables. Then we rank features based on their distinguishing power to classify emails into the two classes using standard
metrics: Chi-Square Statistics [187] and Mutual Information [101]. As shown in Table 3.3, consistently, “receiver authentication method” is the most important factor, followed by the “spoofed sender address”. Note that this analysis only compares the relative importance of factors in our experiment. We are not trying to reverse-engineer the complete defense system, which requires analyzing more features.

Discussion. It takes both the sender and the receiver to make a reliable email authentication. When one of them fails to do their job, there is a higher chance for the forged email to reach the inbox. In addition, email providers tend to prioritize email delivery over security. When an email fails the authentication, most email providers (including Gmail and iCloud) would still deliver the email as long as the policy of the spoofed domain is not “reject”. Based on the earlier measurement result (§3.3), only 13% of the 1 million domains have set a “reject” or “hard fail” policy, which leaves plenty of room for attackers to perform spoofing.

Our analysis also revealed a vulnerability in two email services (sapo.p and runbox.com), which would allow an attacker to send spoofing emails through the email provider’s IP. Since this is a different threat model, we discuss the details of this vulnerability in Appendix B.

3.5.3 Email Clients and Security Indicators

For emails that reached the user inbox, we next examine the security indicators on email interfaces to warn users. Again the results represent the state of email services as of January 2018.

Web Client. We find that only 6 email services have displayed security indicators on forged emails including Gmail, and protonmail, naver, mail.ru, 163.com and 126.com (Figure 3.5 (a)–(e)). Other email services display forged emails without any visual alert (e.g., Yahoo Mail, iCloud). Particularly, Gmail and Google Inbox are from the same company, but the web version of Google Inbox has no security indicator. Gmail’s indicator is a “question mark” on the sender’s icon. Only when users move the mouse over the image, it will show the following message: “Gmail could not verify that <sender> actually sent this message (and not a spammer)”.

Mobile Client. Even fewer mobile email apps have adopted security indicators. Out of the 28 email services with a dedicated mobile app, only 4 services have mobile security indicators including naver, protonmail, Gmail, and google inbox. The other services removed the security indicators for mobile users. Compared to the web interface, mobile apps have very limited screen size. Developers often remove “less important” information to keep a clean interface. Unfortunately, the security indicators are among the removed
3.5. Spoofing Experiment Results

We find that attackers can trigger misleading UI elements to make the forged email look realistic.

**Spoofing an Existing Contact.** When an attacker spoofs an existing contact of the receiver, the forged email can automatically load misleading UI elements such as the contact’s photo, name card, or previous email conversations. We perform a quick experiment as follows: First, we create an “existing contact” (contact@vt.edu) for each receiver account in the 35 email services, and add a name, a profile photo and a phone number (if allowed). Then we spoof this contact’s address (contact@vt.edu) to send forged emails. Table 3.4 shows the 25 email providers that have misleading UIs. Example screenshots are shown in Appendix C. We believe that these designs aim to improve the usability of the email service by providing the context for the sender. However, when the sender address is actually spoofed, these UI elements would help attackers to make the forged email look more authentic.

In addition, spoofing an existing contact allows forged emails to penetrate new email providers. For example, Hotmail blocked all the forged emails in Table 3.2. However, when we spoof an existing contact, Hotmail delivers the forged email to the inbox and adds a special warning sign as shown in Figure 3.5(i).

### Table 3.4: Misleading UI elements when the attacker spoofs an existing contact. (*) indicates web interface only. (†) indicates mobile only.

<table>
<thead>
<tr>
<th>Misleading UI</th>
<th>Email Providers (25 out of 35)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sender Photo (6)</td>
<td>G-Inbox, Gmail, zoho, icloud*, gmx†, mail.com†</td>
</tr>
<tr>
<td>Name Card (17)</td>
<td>yahoo, hotmail, tutanota, seznam.cz, fastmail, gmx, mail.com, Gmail*, sina*, junoo*, aol*, 163.com†, 126.com†, yeah.net†, sohu†, naver†, zoho†</td>
</tr>
<tr>
<td>Email History (17)</td>
<td>hotmail, 163.com, 126.com, yeah.net, qq, zoho, mail.ru, yahoo*, Gmail*, sina*, naver*, op.pl*, interia.pl*, daum.net*, gmx.com*, mail*, inbox.lv*</td>
</tr>
</tbody>
</table>

Finally, we check emails using third-party clients including Microsoft Outlook, Apple Mail, and Yahoo Web Mail. We test both desktop and mobile versions, and find that none of them provide security indicators for the forged emails.

3.5.4 Misleading UI Elements
Chapter 3. Email Spoofing Attacks

3.6 Effectiveness of Security Indicators

As an end-to-end study, we next examine the last hop — how users react to spoofing emails. Our result so far shows that a few email providers have implemented visual security indicators on the email interface to warn users of the forged emails. In the following, we seek to understand how effective these security indicators are to improve user efficacy in detecting spoofed phishing emails.
3.6. Effectiveness of Security Indicators

3.6.1 Experiment Methodology

To evaluate the effectiveness of security indicators, we design an experiment where participants receive a phishing email with a forged sender address. By controlling the security indicators on the interface, we assess how well security indicators help users to handle phishing emails securely.

Implementing this idea faces a key challenge, which is to capture the realistic user reactions to the email. Ideally, participants should examine the phishing email without knowing that they are in an experiment. However, this leads to practical difficulties to set up the user study and obtain the informed user consent up front. To this end, we introduce deception to the study methodology. At the high level, we use a distractive task to hide the true purpose of the study before and during the study. Then after the study is completed, we debrief the users to obtain the informed consent. Working closely with our IRB, we have followed the ethical practices to conduct the phishing test.

**Procedure.** We frame the study as a survey to understand users’ email habits. The true purpose is hidden from the participants. This study contains two phases. Phase 1 is to set up the deception and phase 2 carries out the phishing experiment.

*Phase 1:* The participants start by entering their own email addresses. Then we immediately send the participants an email and instruct the participants to check this email from their email accounts. The email contains a tracking pixel (a 1×1 transparent image) to measure if the email has been opened. After that, we ask a few questions about the email (to make sure they actually opened the email). Then we ask other distractive survey questions about their email usage habits. Phase 1 has three purposes: (1) to make sure the participants actually own the email address; (2) to test if the tracking pixel works, considering some users may configure their email service to block images and HTML; (3) to set up the deception. After phase 1, we give the participants the impression that the survey is completed (participants get paid after phase 1). In this way, participants would not expect the second phishing email.

*Phase 2:* We wait for 10 days and send the phishing email. The phishing email contains a benign URL pointing to our own server to measure whether the URL is clicked. In addition, the email body contains a tracking pixel to measure if the email has been opened. As shown in Figure 3.7, we impersonate the tech-support of Amazon Mechanical Turk (support@mturk.com) to send the phishing email that informs some technical problems. This email actually targeted our own institution before. The phishing email is only sent to users whose email service is not configured to block HTML or tracking pixels (based on phase 1).

We wait for another 20 days to monitor user clicks. After the study, we send a debriefing email which explains the true purpose of the experiment and obtains the informed consent. Participants can withdraw their data anytime. By the time of our submission, none of the users have requested to withdraw their data.
Chapter 3. Email Spoofing Attacks

Figure 3.7: The phishing email screenshot.

Security Indicators. Based on our previous measurement results, most email services adopted text-based indicators (Figure 3.5(b)-(i)). Even GMail’s special indicator (Figure 3.5(a)) will display a text message when users move the mouse over. To this end, we use the text-based indicator and make two settings, namely with security indicator and without security indicator. For the group without security indicator, we recruit users from Yahoo Mail. We choose Yahoo Mail users because Yahoo Mail is the largest email service that has not implemented any security indicators. For the comparison group with security indicator, we still recruit Yahoo Mail users for consistency, and add our own security indicators to the interface. More specifically, when sending emails, we can embed a piece of HTML code in the email body to display a text-based indicator. This is exactly how most email providers insert their visual indicators in the email body (except for Gmail).

In phase2, we cannot control if a user would use the mobile app or the website to read the email. This is not a big issue for Yahoo Mail users. Yahoo’s web and mobile clients both render HTML by default. The text-based indicator is embedded in the email body by us,
3.6. Effectiveness of Security Indicators

which will be displayed consistently for both web and mobile users (confirmed by our own tests).

**Recruiting Participants.** To collect enough data points from phase 2, we need to recruit a large number of users given that many users may not open our email. We choose Amazon Mechanical Turk (MTurk), the most popular crowdsourcing platform to recruit participants. MTurk users are slightly more diverse than other Internet samples as well as college student samples. Using Amazon Mechanical Turk may introduce biases in terms of the user populations. However, the diversity is reportedly better than surveying the university students [70]. To avoid non-serious users, we apply the screening criteria that are commonly used in MTurk [75, 133]. We recruit users from the U.S. who have a minimum Human Intelligence Task (HIT) approval rate of 90%, and more than 50 approved HITs.

In total, we recruited $N = 488$ users from MTurk: 243 users for the “without security indicator” setting, and another 245 users for the “with security indicator” setting. Each user can only participate in *one setting for only once* to receive $0.5. In the recruiting letter, we explicitly informed the users that we need to collect their email address. This may introduce self-selection biases: we are likely to recruit people who are willing to share their email address with our research team. Despite the potential bias, that the resulting user demographics are quite diverse: 49% are male and 51% are female. Most participants are 30–39 years old (39.1%), followed by users under 29 (31.8%), above 50 (14.5%), and 40–49 (14.5%). Most of the participants have a bachelor degree (35.0%) or a college degree (33.8%), followed by those with a graduate degree (20.7%) and high-school graduates (10.5%).

**Ethics Guidelines.** Our study received IRB approval, and we have taken active steps to protect the participants. First, only benign URLs are placed in the emails which point to our own server. Clicking on the URL does not introduce practical risks to the participants or their computers. Although we can see the participant’s IP, we choose not to store the IP information in our dataset. In addition, we followed the recommended practice from IRB to conduct the deceptive experiment. In the experiment instruction, we omit information only if it is absolutely necessary (*e.g.*, the purpose of the study and details about the second email). Revealing such information upfront will invalidate our results. After the experiment, we immediately contact the participants to explain our real purpose and the detailed procedure. We offer the opportunity for the participants to opt out. Users who opt-out still get the full payment.

3.6.2 Experiment Results

We analyze experiment results to answer the following questions. First, how effective are security indicators in protecting users? Second, how does the impact of security indicators vary across different user demographics?
### Chapter 3. Email Spoofing Attacks

#### Table 3.5: User study statistics.

<table>
<thead>
<tr>
<th>Phase</th>
<th>Users</th>
<th>w/o Indict.</th>
<th>w/ Indict.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phase 1</td>
<td>All Participants</td>
<td>243</td>
<td>245</td>
</tr>
<tr>
<td></td>
<td>Not Block Pixel</td>
<td>176</td>
<td>179</td>
</tr>
<tr>
<td>Phase 2</td>
<td>Opened Email</td>
<td>94</td>
<td>86</td>
</tr>
<tr>
<td></td>
<td>Clicked URL</td>
<td>46</td>
<td>32</td>
</tr>
<tr>
<td>Click Rate</td>
<td>Overall</td>
<td>26.1%</td>
<td>17.9%</td>
</tr>
<tr>
<td></td>
<td>After Open Email</td>
<td>48.9%</td>
<td>37.2%</td>
</tr>
</tbody>
</table>

#### Table 3.6: User study statistics for different user-agents.

<table>
<thead>
<tr>
<th>Users</th>
<th>w/o Indicator Desktop</th>
<th>Mobile</th>
<th>w/ Indicator Desktop</th>
<th>Mobile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Opened Email</td>
<td>45</td>
<td>49</td>
<td>41</td>
<td>45</td>
</tr>
<tr>
<td>Clicked URL</td>
<td>21</td>
<td>25</td>
<td>15</td>
<td>17</td>
</tr>
<tr>
<td>Click Rate</td>
<td>46.7%</td>
<td>51.0%</td>
<td>36.6%</td>
<td>37.8%</td>
</tr>
</tbody>
</table>

**Click-through Rate.** Table 3.5 shows the statistics for the phishing results. For phase-2, we calculate two click-through rates. First, out of all the participants that *received* the phishing email, the click-through rate with security indicator is $32/179=17.9\%$. The click-through rate without security indicator is higher: $46/176=26.1\%$. However, this comparison is not entirely fair, because many users did not open the email, and thus did not even see the security indicator at all.

In order to examine the impact of the security indicator, we also calculate the click-through rate based on users who *opened* the email. More specifically, we sent phishing emails to the 176 and 179 users who did not block tracking pixels, and 94 and 86 of them have opened the email. This returns the email-opening rate of 53.4\% and 48.9\%. Among these users, the corresponding click-through rates are 48.9\% (without security indicator) and 37.2\% (with security indicator) respectively. The results indicate that security indicators have a positive impact to reduce risky user actions. When the security indicator is presented, the click rate is *numerically* lower compared to that without security indicators. The difference, however, is not very significant (Fisher’s exact test $p = 0.1329$). We use Fisher’s exact test instead of the Chi-square test due to the relatively small sample size. The result suggests that the security indicator has a moderately positive impact.

**User Agents.** In our experiment, we have recorded the “User-Agent” when the user opens the email, which helps to infer the type of device that a user was using to check the email. Recall that no matter what device the user was using, our security indicator (embedded in the email body) will show up regardless. Table 3.6 shows that mobile users are more likely to click on the phishing link compared with desktop users, but the difference is not significant.
3.7. Discussion

In this section, we summarize our results and discuss their implications for defending against email spoofing and broadly spear phishing attacks. In addition, we discuss the new changes made by the email services after our experiment, and our future research directions.
3.7.1 Implications of Our Results

**Email Availability vs. Security.** Our study shows many email providers choose to deliver a forged email to the inbox even when the email fails the authentication. This is a difficult trade-off between security and email availability. If an email provider blocks all the unverified emails, users are likely to lose their emails (e.g., from domains that did not publish an SPF, DKIM or DMARC record). Losing legitimate emails is unacceptable for email services which will easily drive users away.

The challenge is to accelerate the adoption of SPF, DKIM and DMARC. Despite the efforts of the Internet Engineering Task Force (IETF), these protocols still have limitations to handle special email scenarios such as mail forwarding and mailing lists, creating further obstacles to a wide adoption [104, 156, 173]. Our measurement shows a low adoption rate of SPF (44.9%) and DMARC (5.1%) among the Internet hosts. From the email provider’s perspective, the ratio of unverified inbound emails is likely to be lower since heavy email-sending domains are likely to adopt these protocols. According to the statistics from Google in 2015 [114], most inbound emails to Gmys have either SPF (92%) or DKIM (83.0%), but only a small portion (26.1%) has a DMARC policy. This presents an on-going challenge since spear phishing doesn’t require a large volume of emails to get in. Sometimes one email is sufficient to breach a target network.

**Countermeasures and Suggestions.** First and foremost, email providers should consider adopting SPF, DKIM and DMARC. Even though they cannot authenticate all the incoming emails, these protocols allow the email providers to make more informed decisions. Further research is needed to ease the deployment process and help to avoid disruptions to the existing email operations [97].

In addition, if the email providers decide to deliver an unverified email to the inbox, we believe it is necessary to place a security indicator to warn users based on our user study results. A potential benefit is that the security indicator can act as a forcing function for sender domains to configure their SPF/DKIM/DMARC correctly.

Third, we argue that email providers should make the security indicators *consistently* for different interfaces. Currently, mobile users are exposed to a higher-level of risks due to the lack of security indicators. Another example is that Google Inbox (web) users are less protected compared to users that use Gmail’s interface.

Finally, the misleading UI elements such as “profile photo” and “email history” should be disabled for emails with unverified sender addresses. This should apply to both spoofing an existing contact and spoofing users in of same email provider. So far, we have communicated our results with the Gmail team and provided the suggestions on improving the current security indicators. We are in the process of communicating with other email providers covered in our study.
3.7. Discussion

New Protocols BIMI and ARC. Recently, new protocols are developed to enhance spoofing detection. For example, BIMI (Brand Indicators for Message Identification) is a protocol built on DMARC. After confirming the authenticity of the email sender via DMARC, the email client can display a BIMI logo as a security indicator for the sender brand. This means emails with a BIMI logo are verified, but those without the BIMI logo are not necessarily malicious.

ARC (Authenticated Received Chain) is an under-development protocol that works on top of SPF, DKIM and DMARC. ARC aims to address the problems caused by mail forwarding and mailing lists. For example, when an email is sent through a mailing list, the email sending IP and the email content might be changed (e.g., adding a footer) which will break SPF or DKIM. ARC proposes to preserve the email authentication results through different sending scenarios. For both ARC and BIMI, they are likely to face the same challenge to be widely adopted just like DMARC (standardized in 2015).

3.7.2 UI Updates from Email Services

A few email services have updated their user interfaces during January – June in 2018. Particularly, after we communicate our results to the Gmail team, we notice some major improvements. First, when we perform the same-domain spoofing (i.e., spoofing a Gmail address), in addition to the question-mark sign, there is a new warning message added to the email body as shown in Figure 3.9. Second, the new mobile Gmail app no longer displays the “misleading” profile photos on unverified messages (regardless spoofing existing contact or the same-domain account). The same changes are applied to the new Google Inbox app too. However, the mobile clients are still not as informative as the web version. For example, there is no explanation message on the question-mark sign on the mobile apps. In addition, the new warning message (Figure 3.9) has not been consistently added to the mobile apps either.

Inbox.lv has launched its mobile app recently. Like its web version, the mobile app does not provide a security indicator. However, the UI of the mobile app is simplified which no longer loads misleading elements (e.g., profile photos) for unverified emails. Yahoo Mail and Zoho also updated their web interfaces but the updates were not related to security features.

3.7.3 Open Questions & Limitations

Open Questions. It is unlikely that the email spoofing problem can quickly go away given the slow adoption rate of the authentication protocols. Further research is needed to design more effective indicators to maximize its impact on users. Another related question is how to maintain the long-term effectiveness of security indicators and overcome the “warning fatigue” [69]. Finally, user training/education will be needed to teach users how to interpret
the warning message, and handle questionable emails securely. For security-critical users (e.g., journalists, government agents, military personnel), an alternative approach is to use PGP to prevent email spoofing \cite{134}. Extensive work is still needed to make PGP widely accessible and usable for the broad Internet population \cite{136,201}.

**Study Limitations.** Our study has a few limitations. First, our measurement only covers public email services. Future work will explore if the conclusion also applies to non-public email services. Second, while we have taken significant efforts to maintain the validity of the phishing test, there are still limits to what we can control. For ethical considerations, we cannot fully scale-up the experiments beyond the 488 users, which limited the number of variables that we can test. Our experiment only tested a binary condition (with or without a security indicator) on one email content. Future work is needed to cover more variables to explore the design space such as the wording of the warning messages, the color and the font of the security indicator, the phishing email content, and the user population (e.g., beyond the MTurk and Yahoo Mail users). Finally, we use “clicking on the phishing URL” as a measure of risky actions, which is still not the final step of a phishing attack. However, tricking users to give way their actual passwords would have a major ethical implication, and we decided not to pursue this step.

### 3.8 Related Work

**Email Confidentiality, Integrity and Authenticity.** SMTP extensions such as SPF, DKIM, DMARC and STARTTLS are used to provide security properties for email transport. Recently, researchers conducted detailed measurements on the server-side usage of these protocols \cite{114,131,148,154}. Unlike prior work, our work shows an end-to-end view and demonstrate the gaps between server-side spoofing detection and the user-end notifications. Our study is complementary to existing work to depict a more complete picture.

**Email Phishing.** Prior works have developed phishing detection methods based on features extracted from email content and headers \cite{108,112,127,149,212,243}. Phishing detection is different from spam filtering \cite{246} because phishing emails are not necessarily sent in bulks \cite{294} but can be highly targeted \cite{146}. Other than spoofing, attackers may also apply typosquatting or unicode characters \cite{62} to make the sender address appear similar (but not identical) to what they want to impersonate. Such sender address is a strong indicator of phishing which has been used to detect phishing emails \cite{178,183}. Another line of research focuses on the phishing website, which is usually the landing page of the URL in a phishing email \cite{102,142,284,298,321,322}.

Human factors (demographics, personality, cognitive biases, fatigue) would affect users response to phishing \cite{98,138,159,190,229,235,259,289,295,299}. The study results have been used to facilitate phishing training \cite{296}. While most of these studies use the “role-
playing” method, where users read phishing emails in the simulated setting. There are rare exceptions \[159, 229\] where the researchers conducted a real-world phishing experiment. Researchers have demonstrated the behavioral differences in the role-playing experiments with reality \[256\]. Our work is the first to examine the impact of security indicators on phishing emails using realistic phishing tests.

**Visual Security Indicators.** Security Indicators are commonly used in web or mobile browsers to warn users of unencrypted web sessions \[125, 164, 203, 269\], phishing web pages \[109, 116, 299, 315\], and malware sites \[64\]. Existing work shows that users often ignore the security indicators due to a lack of understanding of the attack \[299\] or the frequent exposure to false alarms \[181\]. Researchers have explored various methods to make security UIs harder to ignore such as using attractors \[82, 83, 84\]. Our work is the first to measure the usage and effectiveness of security indicators on forged emails.

### 3.9 Conclusion

Through extensive end-to-end measurements and real-world phishing tests, our work reveals a concerning gap between the server-side spoofing detection and the actual protection on users. We demonstrate that most email providers allow forged emails to get to user inbox, while lacking the necessary warning mechanism to notify users (particularly on mobile apps). For the few email services that implemented security indicators, we show that security indicators have a positive impact on reducing risky user actions under phishing attacks but cannot eliminate the risk. We hope the results can help to draw more community attention to promoting the adoption of SMTP security extensions, and developing effective security indicators for the web and mobile email interfaces.
Chapter 4

The Adoption of Anti-Spoofing Protocols in Email Systems

4.1 Introduction

Phishing attack has been a persistent threat to the Internet. Recently, this threat has been significantly escalated due to its heavy involvement in massive data breaches [288], ransomware outbreaks [207], and even political campaigns [122]. For example, spear phishing emails have been used in nearly half of the recent 2000 data breaches, responsible for leaking billions of data records [288].

Email spoofing is a critical step in phishing attacks where the attacker impersonates someone that the victim knows or trusts. By spoofing the email address of a reputable organization or a close friend, the attacker has a better chance to deceive the victim [159]. To prevent spoofing, there has been an active effort since the early 2000 to develop, promote, and deploy anti-spoofing protocols. Protocols such as SPF [173], DKIM [104], and DMARC [204] have become the Internet standards, allowing email receivers to verify the sender’s identity.

Despite these efforts, however, sending spoofing emails is still surprisingly easy today. As an example, Figure 4.1 shows a spoofing email where the sender address is set to the domain of the U.S. Citizenship and Immigration Services (USCIS). We crafted and sent this email to our own account in Yahoo (as the victim), and it successfully reached the inbox without triggering any warnings. This is not a coincident as email spoofing is still widely used in real-world phishing attacks [122, 240, 288].

The real question is, why email spoofing is still possible after years of efforts spent on the defense. In 2015, two measurement studies [114, 131] show that the adoption rates of anti-spoofing protocols are still low. Among Alexa top 1 million domains, only 40% have adopted SPF and only 1% have DMARC. We repeated the same measurement methodology recently in 2018, and found that the adoption rates were not significantly improved (SPF 44.9%, DMARC 5.1%). It is not yet clear what causes the slow progress of adopting anti-spoofing solutions.

In this chapter, we seek to understand why anti-spoofing protocols are not widely adopted, particularly from email providers’ perspectives. We planned to conduct a user study with email administrators from different institutions, which turned out to be challenging. Part
4.1. Introduction

Figure 4.1: A spoofing email that impersonates the U.S. Citizenship and Immigration Services (USCIS). We acted as the attacker and sent this email to our own account. The email arrived in the inbox without triggering any alert.

of the reason is that the candidate pool is small. People who can provide insights for our questions need to have extensive experience managing real-world email services. In addition, email administrators often hesitate (or are not allowed) to share details about their anti-phishing/spoofing solutions. To these ends, we send our user study requests to 4000 email administrators of Alexa top domains. We eventually received responses from $N = 9$ administrators from various organizations (universities, payment services, online community websites) who agree to answer open questions either online or through in-person interviews.

Our results show that email administrators are aware of and also concerned about the technical weaknesses of SPF, DKIM and DMARC. Based on interview results and by reading the protocol specifications, we summarize 6 key weaknesses across the three protocols. These technical weaknesses either allow spoofing emails to bypass the authentication check or block legitimately forwarded emails. The general perception is that these protocols are “helpful”, but “cannot solve the spoofing problem completely”.

In addition, the email administrators believe that the slow adoption of the protocols is primarily due to the lack of a critical mass. Like many network protocols, the benefits of the anti-spoofing protocols come into existence only if a large number of Internet domains start to adopt the protocols to publish their authentication records. Currently, the incentive of adoption is not strong, especially for Internet domains that don’t host emails services (which can still be spoofed).

Finally, the email administrators pointed out the practical challenges to deploy the protocols, particularly, for organizations that use cloud-based email services and large organizations that have many dependent services. Our study participants also shared their thoughts on the possible solutions moving forward. One interesting direction is to improve the current email user interface to support security indicators, and educate users to proactively check
Chapter 4. The Adoption of Anti-Spoofing Protocols in Email Systems

Table 4.1: User study participants: 9 email administrators. U8 requested to conceal the institution type, and thus we keep it as “anonymous”. For each of their email services, we also measured whether the email domain published the DNS authentication records (as the sender) and whether the domain authenticate incoming emails (as the receiver). “✓” means the mail server has adopted SPF/DKIM/DMARC. “✗” means the mail server did has not adopted SPF/DKIM/DMARC. “/” means not applicable. Note that we could not obtain a mail server’s DKIM record from the DNS since the selector information is not public.

In summary, our work makes three contributions.

• First, we extracted and categorized 6 technical weaknesses in the existing anti-spoofing protocol designs based on our user study (and the protocol specifications). The result provides the taxonomy of the problem.

• Second, through the user study, we provide new insights into the perceived values and concerns of anti-spoofing protocols from email providers’ perspectives. These results shed light to the reasons behind the slow adoption of SPF, DKIM, and DMARC, pointing out the directions of improvement moving forward.

• Third, we discuss the key implication of the results to protocol designers, email providers, and users. We discuss the possible solutions at the user-end to make up for the defective server-side authentication.

4.2 User Study Methodology

In this chapter, we conduct an exploratory study to understand the adoption of anti-spoofing protocols. We qualitatively look into the perceptions of email administrators towards existing anti-spoofing protocols. We primarily focus on two aspects: the perceived usefulness (PU) and the perceived ease-of-use (PEOU), which are the two most important factors for general technology adoption [186, 285, 286]. Below, we introduce the methodology of our user study.
4.2. User Study Methodology

Table 4.2: Technical weaknesses of SPF, DKIM and DMARC.

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Weakness</th>
<th>Problem Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPF</td>
<td>P1. Alignment</td>
<td>The verified sender can be different from the one displayed to users.</td>
</tr>
<tr>
<td></td>
<td>P2. Mail forward</td>
<td>A forwarded email by default cannot pass the SPF test.</td>
</tr>
<tr>
<td></td>
<td>P3. Mailing list</td>
<td>Emails sent to a mailing list by default cannot pass the SPF test.</td>
</tr>
<tr>
<td>DK.</td>
<td>P4. Alignment</td>
<td>The sender domain that signed DKIM can be different from the one user sees.</td>
</tr>
<tr>
<td></td>
<td>P5. Mailing list</td>
<td>Mailing lists often modify the email content, which will fail the DKIM test.</td>
</tr>
<tr>
<td>DM.+SPF</td>
<td>P2. Mail forward</td>
<td>A forwarded email by default fails the SPF, and thus fails DMARC.</td>
</tr>
<tr>
<td></td>
<td>P3. Mailing list</td>
<td>Emails sent to a mailing list cannot pass SPF and DMARC at the same time.</td>
</tr>
<tr>
<td>DMARC+DKIM</td>
<td>P5. Mailing list</td>
<td>Mailing lists often modify the email content, which will fail the DKIM test.</td>
</tr>
<tr>
<td>DM.+SPF+DK.</td>
<td>P5. Mailing list</td>
<td>SPF always fails; DKIM will fail if the mailing list modifies email content.</td>
</tr>
</tbody>
</table>

The biggest challenge of our user study is to recruit participants. We need to recruit participants who have real-world experience of operating an email service and/or deploying anti-spoofing protocols. This narrows down the candidate pool to a small and highly specialized user population. In addition, real-world email administrators are often reluctant to share due to the sensitivity of the topic. For many companies and organizations, details about their phishing/spoofing detection systems are non-disclosable.

To address these challenges, we sent our user study requests to a large number of email administrators. More specifically, we contacted the email administrators of Alexa top 4000 domains. In the user study request, we ask about their preferred ways of participation (e.g., survey, phone interviews) and the level of details they feel comfortable to share. In total, we recruit \( N = 9 \) email providers from different organizations. 7 participants agree to fill in a survey with “open questions” and 2 participants agree to do an in-person interview. In Table 4.1, we list the 9 email administrators and the type of their institutions and organizations. Note that U8 requested to conceal the institution-specific information, and thus we keep it as “anonymous”. This small-scale but in-depth user study seeks to provide useful qualitative results and new insights from protocol users’ perspectives.

To provide the context for each email service that the participant manages, we also performed a quick measurement as shown in Table 4.1. We measured whether the email domain published the corresponding authentication records in DNS (as the sender) and whether the domain performed authentication checks on the incoming emails (as the receiver). As mentioned in 2.1.2, we cannot measure whether an email domain has published the DKIM public key without knowing its selector (marked with “/”). We observe that most of the email services perform all three authentication checks on incoming emails (7 out of 8) and one email service checks DKIM only. However, when acting as the sender domain, only 3 email services published both SPF and DMARC records to the DNS.

For the interview and survey participants, we use the same list of open questions. The difference is that we can ask follow-up questions to the interview participants, but not the survey participants. At the high-level, the open questions fall into the following themes. First, we ask the participants to comment on the email spoofing problem and how they usually detect spoofing attempts. Second, we ask the participants to comment on the value
and potential weaknesses of SPF, DKIM and DMARC. Third, we ask about their personal perceptions towards the under-adoption of anti-spoofing protocols and the possible reasons. Fourth, we ask the participants to comment on the possible solutions moving forward to the email spoofing problem.

The survey participants answer the open questions using an online survey website that we set up. The interview participants then have a face-to-face interview session for 45 to 60 minutes. Our study is approved by IRB. We ensure that all the data are properly anonymized and securely stored.

4.3 User Study Results

In the following, we discuss our user study results regarding the values and concerns of SPF, DKIM and DMARC, and the possible reasons behind their slow adoption. We group the results into 6 high-level topics.

4.3.1 Technical Defects of the Protocols

Email administrators have acknowledged the values of adoption these protocols. However, the most discussed topics are still the technical flaws in SPF, DKIM and DMARC. In the following, we categorize and summarize 5 key weaknesses of the anti-spoofing protocols based on the user study results, as shown in Table 4.2. We have validated these weaknesses by (1) reading and the protocol specifications, and (2) deploying SPF, DKIM and DMARC on our own mail server and running proof-of-concept experiments.

Identifier Alignment (P1, P4). SPF and DKIM both have the problem of “identifier alignment”. It means that the sender email address that user sees can be different from the address that is actually used to do perform authentication. Figure 4.2 shows an example for SPF. For SPF, the authentication focuses on the “Return-Path” and examines whether the sender’s IP is listed in the “Return-Path” domain’s SPF record. An attacker can set the “Return-Path” domain to her own domain and set her SPF record to pass the authentication. However, what the receiving user sees on the email interface is set by the “From” field. Since SPF does not require the two domains to be the same, then the spoofing email can pass the SPF check while displaying the impersonated address to users. DKIM has a similar problem given that the domain to sign the email with the DKIM key can be different from the domain on the “Return-Path”. DMARC helps to revolve the problem by enforcing the alignment of the identifiers.

Mail Forwarding (P2). Mail forwarding is a problem for SPF. Mail forwarding means one email service automatically forwards emails to another email service. A common scenario is that university students often configure their university email service to forward all their
4.3. User Study Results

Figure 4.2: SPF: SPF test is based on the domain of “Return-Path”, which can be different from the domain that the user sees (the “From” field).

emails to Outlook or Gmail. During Mail forwarding, the email metadata (e.g., “Return-Path”) remains unchanged. SPF will fail after mail forwarding because the forwarder’s IP will not match the original sender’s SPF record. DMARC cannot solve the mail forwarding problem of SPF.

Mailing List (P3, P5). Mailing list is a major problem for both SPF and DKIM. When a message is sent to a mailing list, the mailing list will “broadcast” the message to all the subscribers. This is a similar process as mail forwarding. During this process, the mailing list’s IP will become the sender IP, which is different from the original sender’s IP. This will lead to SPF failure.

Mailing lists will cause trouble for DKIM because most mailing lists modify the email content before broadcasting it to the subscribers. The common modification is to add a “footer” with the name of the mailing list and a link for un-subscription. Tampering the email content will cause DKIM failure.

DMARC cannot solve the mailing list problem. For mailing lists, DMARC+SPF will be sure to fail: if the “Return-Path” is modified, DMARC will fail due to the misalignment of identifiers; if the “Return-Path” is unmodified, SPF will fail due to the IP mismatch. For DMARC+DKIM, it will fail if the mailing list still has to modify the email content.

In particular, U7 pointed out the problem of DKIM beyond just the mailing list problem. U7 stated that DKIM was too sensitive to “benign” changes to the email content such as line rewrapping and URL expansion. These operations that are very common in email services (sometimes for usability purposes), but can easily lead to invalid signatures. The sensitivity of DKIM also discourages email administrators from deploying DMARC (which need to work with DKIM).

“U7: DKIM is inherently flawed because semantically meaningless changes to a message can render the signature invalid. For example, the relaxed body canonicalization algorithm is sensitive to line rewrapping, which will invalidate the
signature without changing the semantic content of the message. Flaws like this make DKIM signatures fragile, reducing the utility of DKIM and thus lessening the priority of its deployment.”

“U7: The fragility of DKIM also affects the utility of DMARC, and thus reducing the priority of its deployment as well.”

4.3.2 A Lack of Critical Mass

Email administrators mentioned that there had not been a global consensus that SPF, DKIM or DMARC should be the ultimate solution to stop spoofing. Part of the reason is these protocols are struggling to support common email scenarios such as mail forwarding. Due to the technique weaknesses, the general perception is that SPF, DKIM and DMARC are “helpful” but “cannot solve the spoofing problem completely”. U2 mentioned that potential adopters could be are waiting to see whether enough people would eventually get on board.

“U2: It is not the final answer that the industry picked up yet. I felt at this point that enough people haven’t really adopted it, it’s not worth for me to set it up.”

This reflects a typical bootstrapping challenge, where a “critical mass” is needed in order to facilitate a self-sustaining adoption process [231]. A related notion is the Network Externalities (or net effect) [80, 167]. Network externalities mean that an individual adopter can add the value for other people to adopt the same technology. In other words, when more users adopt the same protocol, the value of the protocol to each user will also increase [251]. For anti-spoofing protocols, if more domains publish their SPF/DKIM/DMARC records, it makes easier for other email providers to detect spoofing emails.

4.3.3 Benefits Not Significantly Overweight Costs

Email administrators then discussed the deeper reasons for the lack of critical mass. U1 pointed out that the protocol adopter does not directly benefit from publishing their SPF, DKIM or DMARC records in the DNS. Instead, these DNS records mainly help other email services to verify incoming emails and protect the customers (users) of other email services. Domains that publish the DNS records receive the benefit of a better reputation, which is a relatively vague benefit, particularly for domains that don’t host email services.

“U1: If I am an email provider, I am not motivated to set up SPF, I am motivated to make sure people who have sent (emails) to my customers have set SPF. I am motivated to evaluate it.”
4.3. User Study Results

Figure 4.3: The adoption model for anti-spoofing protocols. For email domains, the cost and benefit changes as more domains adopt the protocol. For non-email domains, the cost and benefit stay constant.
For popular online services (e.g., social networks, banks), however, they are likely to be motivated to publish SPF, DKIM, and DMARC records to prevent being spoofed and maintain their good reputation (U2, U3).

To help to illustrate this challenge, we plot Figure 4.3, which is a modified version of the Ozment-Schechter model [231]. Ozment-Schechter model depicts the general challenge for network protocols to receive a wide adoption. The model argues that only when the benefits to individual adopters overweight the adoption costs will the protocol be widely accepted. For network protocols, the per-user benefits may grow as more users adopt the protocol (net effect) [58]. The costs can be either constant or changing (mostly decreasing) as more users adopt the protocol. We have adapted this model to the email spoofing scenarios and created a separate plot for non-email domains (Figure 4.3(b)).

For email domains (Figure 4.3(a)), when more domains publish their SPF, DKIM or DMARC records, the benefits for each adopter will increase because more incoming emails can be authenticated. Regarding the costs, there will be a constant base cost for deploying the protocol. On top of that, early adopters also need to handle the insecure domains that have not adopted the protocol and those with misconfigurations. With more domains adopting those protocols, there will be fewer emails coming from insecure domains and the cost of insecure domains will drop. However, this cost cannot reach zero due to the technical issues in these protocols as discussed before.

Figure 4.3(b) shows a bigger challenge to motivate non-email domains to publish the SPF/DMARC record. For non-email domains (e.g., office.com), the benefit of publishing the SPF/DMARC record is to prevent attackers from impersonating the non-email domain and helps the non-email domain to maintain a good reputation. The domain administrators publish the SPF/DMARC records to be a good Internet “citizen” and help other email services to detect spoofing emails. However, these benefits are considered indirect and thus relatively weaker (U5, U6). Overall, the cost and benefit model is not in favor of creating a “critical mass” for a wide adoption. The bootstrapping phase is challenging without external enforcement or incentives.

### 4.3.4 Deployment Difficulties in Practice

Even if an email administrator decided to deploy the protocol, there would be other challenges in the way. We summarize the participants’ responses from three aspects: (1) a lack of control on the DNS or even the mail servers, (2) the large number of dependency services, (3) a lack of understanding of the protocol and the deployment difficulties.

First, certain services do not have a control over their DNS record. Publishing SPF/DKIM/DMARC record will incur additional overhead to coordinate with their DNS providers (U1, U4, U9). In addition, many companies and organizations even don’t maintain their own mail servers but rely on cloud-based email services. Using cloud-based email services
is convenient without the need the handle challenging tasks such as spam filtering. The drawback is that the organization need to rely on the cloud email service to deploy the anti-spoofing protocols.

“U1: So we have very limited control over our DNS. Right now, it is just the difficulty of setting up that DNS.”

Another challenge is that the strict enforcement of certain email protocols requires significant efforts for coordination in big institutions. An email system has many dependent services (e.g., marketing tools) distributed in different departments in a big institution. Deploying a new email protocol requires a non-trivial collaboration effort from different departments.

“U7: Strict enforcement requires identifying all the legitimate sources of email using a return address domain. Large, decentralized organizations (e.g. many large universities), will often have organizational units which acquire third-party services involving email, like email marketing tools, without telling central IT. Figuring all this out and putting policies and procedures in place to prevent it is more work than many admins have time for.”

Finally, the participants mentioned that there had been a lack of deep understanding of the anti-spoofing protocols, especially the new protocols such as DMARC. It is difficult to estimate how much effort is needed to deploy and maintain the protocol in practice. U3 particularly mentioned that there is a general perception that deploying anti-spoofing protocols is difficult. Regardless the actual level of the difficulty, the perceived difficulty makes email administrators hesitated to try (U3, U9).

“U3: Many people believe that DKIM is hard, and thus don’t prioritize deploying it ... Many people don’t understand DMARC, how easy it is to deploy, and how effective it is.”

4.3.5 Risks of Breaking the Existing System

Participants have discussed the concerns of breaking the existing email system due to unfamiliarity to the protocol. This is particularly true for DMARC (published in 2015). Email providers need to go through careful testing to make sure the protocol does not block legitimate incoming emails, and their own emails are not blocked by others.

“U2: Probably because it (DMARC) is still in a testing phase and (people) want to see if it is going to work for them. Relatively it (DMARC) is still pretty new for big businesses and such.”
Chapter 4. The Adoption of Anti-Spoofing Protocols in Email Systems

“U5: Domains may fear that they’ve forgotten something and their email may be rejected due to a mistake on their part.”

These concerns also explain why most protocol adopters (as the sender domain) configure a relaxed SPF/DMARC policy \cite{114,131} — even if the authentication failed, email providers can still allow email delivery. \textit{U5} expressed that it was quite often for senders to have misconfigurations. It is easier to not enforce the strict policy than to ask the senders to fix their configurations.

“\textit{U5}: Spam filters are relied upon too heavily and it’s sometimes easier to pull email from the spam folder than ask someone to fix their SPF record and re-send the email.”

### 4.3.6 Solutions Moving Forward

We asked the participants to comment on the possible solutions moving forward. Most of the email administrators believed that automated detection systems (\textit{e.g.}, anti-spoofing protocols, spam filters, virus scanners) were necessary, but could not fully prevent spoofing or phishing. \textit{U1}, \textit{U2}, \textit{U7}, \textit{U8} and \textit{U9} all have mentioned the importance of user education to raise the awareness of spoofing, and training users to check the email authenticity themselves.

“\textit{U7}: There is no one single way. Technological defenses like content filtering of incoming mail (\textit{i.e.} spam and virus filtering), are necessary but not sufficient. There is also a need for rigorous training combined with periodic self-phishing (\textit{e.g.} phishme.com), to raise awareness and identify people who need further training or correction.”

“\textit{U8}: User education is the most important way to protect them. I always ask our users to look for the email that seems suspicious and bring it to my attention. That way we can prevent malicious intention at earliest possible.”

Finally, \textit{U5} expressed the need to have security indicators on the email client. The security indicators are icons or visual cues that are widely used on web browsers to indicate the validity of SSL certificate of websites. A similar email spoofing indicator can be deployed to warn users of emails with unverified sender addresses. In addition, security indicators can also help to high-light the address misalignment of the Return-Path and Mail From fields for emails that bypassed the SPF check.

“\textit{U5}: Add the ability for email clients to warn users similar to the way browsers do when users are either presented with a valid extended SSL cert or no SSL cert at all. May also display the from & reply to addresses making it harder to get around SPF record checking.”
4.4 Discussion

So far, we have explored the challenges for SPF, DKIM and DMARC to receive a wide adoption. Next, we discuss the key implications to protocol designers, email providers, and the end users.

4.4.1 Implications for Protocol Designers and Promoters

Improving the Perceived Usefulness. The security and usability issues in SPF, DKIM and DMARC negatively impact their perceived usefulness. To improve the perceived usefulness, addressing these security and usability issues becomes the first priority. Currently, an IETF group is working on a new protocol called Authenticated Received Chain (ARC) which is expected to address email forwarding problem and the mailing list problem. However, this also adds to the number of protocols that domain owners need to deploy. New protocols will have their own challenges to be accepted. For example, the DMARC protocol, even though incrementally deployable, only achieved a 4.6% adoption rate in the past two years. A useful protocol will still face the challenge to be widely adopted.

Building the Critical Mass. Currently, there is a lack of strong consensus to deploy anti-spoofing protocols. Like many networking protocols, anti-spoofing protocols will provide key benefits only after enough domains start to publish their SPF, DKIM or DMARC records. To bootstrap the adoption and establish a critical mass, external incentive mechanisms are needed. In theory, we can adjust the rewarding function to provide more benefits to early adopters to create a positive net effect. One possible direction is to learn from the promotion of “HTTPS” among websites: modern browsers will display a trusted icon for websites with valid TLS certificates. Similar security indicators can be added to emails with verified sender domains (by SPF, DKIM and DMARC), to incentive domains to publish the corresponding DNS records. In addition, policymakers or major email providers may also consider enforcing certain sensitive domains (e.g., banks, government agencies) to publish their SPF/DKIM/DMARC records to prevent being impersonated. The challenge is how to realize these ideas without disrupting any of the normal operations of the existing email services.

Reducing the Deployment Difficulty. One direction to improve the adoption rate of anti-spoofing protocols is to make it easy to deploy and configure. Our user study reveals two key problems to address. First, more organizations start to use cloud-based email services (e.g., Google G-Suite, Amazon WorkMail, Office 365). Anti-spoofing protocols should be more cloud-friendly for organizations that don’t have full controls on their mail servers. Second, the deployment process should be further simplified and providers email administrators with more controls. The biggest concern from email administrators is that anti-spoofing protocols may reject legitimate emails or get their own emails rejected. One
direction of improvement is to allow the protocol to run in a testing mode (e.g., in DMARC), allowing email administrators to fully assess the impact before real deployment.

4.4.2 Implications for Email Providers

In the short term, email providers are still unlikely to be able to authenticate all the incoming emails. While email providers should act as “good Internet citizens” by publishing their own authentication records, it is also necessary to help to “educate” their users to watch out for spoofing emails. Given the current adoption rate of anti-spoofing protocols (and the relaxed protocol configurations), it is likely that email providers will still have to deliver certain unverified emails to the user inbox. Email providers should act more responsibly by providing the authentication results available for the user to check, or proactively warn users of emails that they are not able to verify. Large email providers such as Gmail and Outlook are already moving towards this direction. Currently, Gmail’s authentication results are available through the webmail interface, but unfortunately not yet available on the mobile app interface. Further research is needed to improve the current mobile email UI to better support security features.

4.4.3 Implications for Users

Given the current situation, users are at the most vulnerable position. Particularly, considering the usability flaws of the existing anti-spoofing protocols, an email that passed the SPF/DKIM checks can still be a spoofed email (e.g., with misaligned addresses). Similarly, emails that failed the SPF/DKIM checks are not necessarily malicious (e.g., forwarded email). To this end, unless the user is fully aware of the authentication details, it is safer for general email users to avoid establishing the trust based on the sender domains. The trustworthiness of the email should be assessed as a whole. It is more reliable to leverage the context of the email exchange, and the external confirmation channels (e.g., calling the sender on the phone) to identify phishing attempts and securely handle critical emails.

4.4.4 Limitations

The scale of the user study is still small, which limits us from producing any statistically significant results. We argue that our contribution is to provide a “qualitative” understanding of the problem space, which lays the groundwork for future quantitative research. For example, one future direction is to conduct surveys to understand what types of domains are more likely to adopt anti-spoofing protocols, and how domain attributes (e.g., service type, popularity, sensitivity) affect the domain owners’ decision.
4.5 Conclusion

In this chapter, we examine why email spoofing is (still) possible in today’s email system. First, our measurement results confirm that anti-spoofing protocols (SPF, DKIM, DMARC) are not widely accepted. Then we qualitatively study the possible reasons for the low adoption rates. By analyzing the discussion threads in IETF and performing user studies with email administrators, we provide a deeper understanding of the perceived value and limitations of anti-spoofing protocols. Our results show that key security and usability limitations are rooted in the protocol design which hurts the perceived usefulness of these protocols. This also makes it difficult to establish a “critical mass” to facilitate a positive net effect for a wider adoption. Moving forward, extensive efforts are needed to address the technical issues in the protocol design and develop external enforcement (or incentives) to bootstrap the protocol adoption. In addition, improved user interfaces are needed for email systems to allow users to proactively check the email authentication results.
Chapter 5

Email Pixel Tracking through Disposable Email Services

5.1 Introduction

An Email address is one of the most important components of personally identifiable information (PII) on the Internet. Today’s online services typically require an email for account registration and password recovery. Unfortunately, email addresses are often unprotected. For example, email addresses used to register online social networks might be collected by malicious third-parties [257], thus exposing users to spam and spear phishing attacks [241]. Massive data breaches, especially those at sensitive services (e.g., Ashley Madison [150]), can expose user footprints online, leading to real-world scandals. In addition, email addresses are often leaked together with passwords [273, 290], allowing malicious parties to link user identities across different services and compromise user accounts via targeted password guessing [292].

As a result, disposable email services have become a popular alternative which allows users to use online services without giving away their real email addresses. From disposable email services, a user can obtain a temporary email address without registration. After a short period of time, the emails will be disposed by the service providers. Users can use this disposable email address for certain tasks (e.g., registering an account on a dating website) without linking their online footprints to their real email addresses (e.g., work or personal email). In this way, potential attacks (e.g., spam, phishing, privacy leakage) will be drawn to the disposable addresses instead of the users’ real email accounts. Disposable email services are highly popular. For example, Guerrilla Mail, one of the earliest services, has processed 8 billion emails in the past decade [24].

While disposable email services allow users to hide their real identities, the email communication itself is not necessarily private. More specifically, most disposable email services maintain a public inbox, allowing any user to access any disposable email addresses at any time [30]. Essentially disposable email services are acting as a public email gateway to receive emails. The “public” nature not only raises interesting questions about the security of the disposable email service itself, but also presents a rare opportunity to empirically collect email data and study email tracking, a problem that is not well-understood.
In this chapter, we have two goals. First, we want to understand what disposable email services are used for in practice, and whether there are potential security or privacy risks involved with using a disposable email address. Second, we use disposable email services as a public “honeypot” to collect emails sent by various online services and analyze email tracking in the wild. Unlike the extensively-studied web tracking [59, 60, 118, 194, 210, 250, 265], email tracking is not well-understood primarily due to a lack of large-scale email datasets. The largest study so far [120] has analyzed emails from 902 “Shopping” and “News” websites. In this chapter, we aim to significantly increase the measurement scale and uncover new tracking techniques.

**Understanding Disposable Email Services.** In this chapter, we collect data from 7 popular disposable email services from October 16, 2017 to January 16, 2018 over three months. By monitoring 56,589 temporary email addresses under popular usernames, we collect in total 2,332,544 incoming email messages sent from 210,373 online services and organizations. We are well aware of the sensitivity of email data. In addition to working with IRB, we also take active steps to ensure research ethics such as detecting and removing PII from the email content and removing personal emails. Our analysis reveals key findings about the usage of disposable email services.

First, there is often a delay to dispose of the incoming emails. Certain services would hold the emails for as long as 30 days, in spite of the claimed 25 minutes expiration time. Second, we find that users are using disposable email addresses to register accounts in a variety of online services. While the vast majority of emails are spam and notifications, we did find a large number of emails (89,329) that are used for account registration, sending authentication code, and even password reset. Third, accounts registered via disposable email addresses are easily hijackable. We find risky usage of disposable email addresses such as registering sensitive accounts at financial services (e.g., PayPal), purchasing bitcoins, receiving scanned documents, and applying for healthcare programs.

**Measuring Email Tracking.** Email tracking involves embedding a small image (i.e., tracking pixel) into the email body to tell a remote server when and where the email is opened by which user. When the email is opened, the email client fetches the pixel and this notifies the trackers. To measure email tracking in the wild, we build a new tool to detect both first-party tracking (where the email sender and the tracker are the same) and third-party tracking (where the email sender and the tracker are different) from the collected email dataset.

We have three key observations. First, email tracking is highly prevalent, especially with popular online services. Out of the 2.3 million emails, 24.6% of them contain at least one tracking link. In terms of sender domains, there are 2,052 sender domains (out of 210K domains in our dataset) ranked within the Alexa top 10K. About 50% of these high-ranked domains perform tracking in their emails. Second, we find that stealthy tracking techniques are universally preferred, either by falsely claiming the size of tracking images in HTML or
hiding the real trackers through redirection. Popular online services are significantly more likely to use “stealthy” tracking techniques. Third, although a small number of trackers stand out in the tracking ecosystem, these trackers are not yet dominating the market. The top 10 email trackers are used by 31.8% of the online domains, generating 12% of the tracking emails. This is different from web tracking where one dominating tracker (i.e., Google) can track user visits of 80% of the online services [199].

**Contributions.** Our work makes three key contributions.

- **First**, we perform the first measurement study on disposable email services by collecting a large-scale dataset (2.3 million emails) from 7 popular services over 3 months.

- **Second**, our analysis provides new insights into the common use cases of disposable email services and uncovers the potential risks of certain types of usage.

- **Third**, we use the large-scale email dataset to empirically measure email tracking in the wild. We show the stealthy tracking methods used by third-party trackers collect data on user identifiers and user actions.

## 5.2 Data Collection

To understand how disposable email services are used, we collect emails that are sent to disposable addresses. First, we describe our data collection process. We then present a preliminary analysis of the dataset. Finally, we discuss the active steps we take to ensure research ethics.

### 5.2.1 Data Crawling Methodology

Since disposable email addresses are public gateways, our method is to set up a list of disposable email addresses and monitor the incoming emails. In this chapter, we primarily focus on user-specified addresses for data collection efficiency. We select a list of “popular” usernames which increases our chance to receive incoming emails. In order to increase our chance of receiving incoming emails, we select a list of “high frequency” usernames. Disposable email addresses under such usernames are often used by multiple users at the same time. In comparison, monitoring randomly-assigned (RA) addresses did not return many incoming emails. For example, in a pilot test, we monitored 5 RA email services (eyepaste.com, getnada.com, mailto.space, mytemp.email, and tempmailaddress.com) for 5 days. We only succeeded in collecting data from getnada.com and all inboxes in other RA services were empty. In total, we scanned 194,054 RA addresses, and collected 1,431 messages from 1,430 inboxes (a hit rate of 0.74%). The reason for the low hit rate is that
5.2. Data Collection

randomly-assigned addresses come from a much larger address space than user-specified ones. Accordingly, in this chapter, we focus on user-specified addresses for data collection.

**Selecting Disposable Email Services.** We spent a few days searching online for “disposable email” and “temporary email” to find popular services. This process mimics how normal users would discover disposable email services. By examining the top 100 entries of the searching results, we find 31 disposable email services (19 UA and 12 RA services\(^1\)). UA services are typically more popular than RA services. For example, the top 5 sites have 4 UA services and 1 RA service. As discussed above, we focus the on services that offer user-specified addresses (UA), and select the top 7 disposable email services as shown in Table 5.2. These services are very popular. For example, guerrillamail.com self-reported that they have processed nearly 8 billion emails in the past decade. mailnesia.com self-reported that they received 146k emails per day. While most of these services only provide the functionality of receiving emails, a few (e.g., guerrillamail.com) also provide the functionality of sending emails. In this work, we only focus on the incoming emails received by the disposable email addresses (to analyze email tracking).

**Selecting Popular Usernames.** We construct a list of popular usernames to set up disposable email addresses. To do so, we analyze 10 large leaked databases (that contain email addresses) from LinkedIn, Myspace, Zoosk, Last.fm, Mate1.com, Neopets.com, Twitter, 000webhost.com, Gmail, Xsplit. These databases are publicly available and have been widely used for password research [105, 196, 273, 279, 287, 290, 292]. By combining the 10 databases, we obtain 430,145,229 unique email addresses and 349,553,965 unique usernames. We select the top 10,000 most popular usernames for our data collection. The top 5 usernames are info, john, admin, mail, and david, where “info” and “david” have been used 800,000 and 86,000 times, respectively.

To confirm that popular usernames are more likely to receive emails, we perform a quick pilot test. We scan all 7 disposable email services, and examine how many addresses under the 10,000 most popular usernames contain incoming emails. From a one-time scan, we find that 8.74% of the popular usernames contain emails at the moment we checked the inbox. As a comparison, we scan a list of random 10,000 usernames and found that only about 1% of addresses contain emails, which confirms our intuition.

**Time Interval for Crawling.** For each disposable email service, we build a crawler to periodically check the email addresses under the top 10,000 usernames. To minimize the impact on the target service, we carefully control the crawling speed and force the crawler to pause for 1 second between two consecutive requests. In addition, we keep a single crawling thread for each service. Under this setting, it would take more than 6 hours to scan all 10K addresses. Considering that certain disposable email services would frequently dispose incoming emails, our strategy is to have an early timeout. Suppose a service keeps an email for \(t\) hours, we design our crawler to stop the current scan once we hit the \(t\)-hour mark, and

\(^1\)Two of the RA services have adopted CAPTCHAs for their sites.
Table 5.1: The expiration time of disposable emails. We show the expiration time claimed on the website and the actual expiration time obtained through measurements.

<table>
<thead>
<tr>
<th>Website</th>
<th>Claimed Time</th>
<th>Actual Time (Min., Avg., Max.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>guerrillamail.com</td>
<td>“1 hour”</td>
<td>1, 1, 1 (hour)</td>
</tr>
<tr>
<td>mailinator.com</td>
<td>“a few hours”</td>
<td>10.5, 12.6, 16.5 (hours)</td>
</tr>
<tr>
<td>temp-mail.org</td>
<td>“25 mins”</td>
<td>3, 3, 3 (hours)</td>
</tr>
<tr>
<td>maildrop.cc</td>
<td>“Dynamic”</td>
<td>1, 1, 1 (day)</td>
</tr>
<tr>
<td>mailnesia.com</td>
<td>“Dynamic”</td>
<td>12.6, 12.8, 13.1 (days)</td>
</tr>
<tr>
<td>mailfall.com</td>
<td>“25 mins”</td>
<td>30, 30, 30 (days)</td>
</tr>
<tr>
<td>mailsac.com</td>
<td>“Dynamic”</td>
<td>19.9, 20.3, 20.7 (days)</td>
</tr>
</tbody>
</table>

Immediately start from the top of the username list. This strategy is to make sure we don’t miss incoming emails to the most popular addresses. Since emails are more likely to hit the top addresses, this strategy allows us to collect more emails with the limited crawling speed.

To set up the early-timeout, we need to measure the email deletion time for each service. We perform a simple experiment: for each service, we first generate 25 random MD5 hash strings as usernames. This is to make sure these addresses are not accidentally accessed by other users during the experiment. Then, we send 25 emails in 5 batches (12 hours apart). In the meantime, we have a script that constantly monitors each inbox to record the message deletion time. In this way, we obtain 25 measurements for each disposable email service.

As shown in Table 5.1, disposable email services often don’t delete emails as quickly as promised. For example, mailfall.com claimed to delete emails in 25 minutes but in actuality, held all the emails for 30 days. Similarly temp-mail.org claimed to delete emails in 25 minutes but kept the emails for 3 hours. This could be an implementation error of the developers or a false advertisement by the service. Many of the services claim that the expiration time is not fixed (which depends on their available storage and email volume). Based on Table 5.1, we only need to apply the early-timeout for temp-mail and guerrillamail to discard lower-ranked usernames, using a timeout of 1 hour and 3 hours respectively.

### 5.2.2 Disposable Email Dataset

We applied the crawler to 7 disposable email services from October 16, 2017 to January 16, 2018 for three months. In total, we collected 2,332,544 email messages sent to monitored email addresses. Our crawler is implemented using Selenium [37] to control a headless browser to retrieve email content. The detailed statistics are summarized in Table 5.2. For 5 of the disposable email services, we can cover all 10K addresses and almost all of them have received at least one email. For the other 2 email services with very a short expiration time (temp-mail and guerrillamail), we focus on an abbreviated version of the popular usernames list. The number of emails per account has a highly skewed distribution. About
5.2. Data Collection

<table>
<thead>
<tr>
<th>Website</th>
<th># Emails</th>
<th>Dispos. Address</th>
<th>Uniq. Sender Address (Domain)</th>
<th>Msgs w/ Sender Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>guerrillamail</td>
<td>1,098,875</td>
<td>1,138</td>
<td>410,457 (190,585)</td>
<td>1,091,230 (99%)</td>
</tr>
<tr>
<td>mailinator</td>
<td>657,634</td>
<td>10,000</td>
<td>27,740 (16,342)</td>
<td>55,611 (8%)</td>
</tr>
<tr>
<td>temp-mail</td>
<td>198,041</td>
<td>5,758</td>
<td>1,748 (1,425)</td>
<td>13,846 (7%)</td>
</tr>
<tr>
<td>maildrop</td>
<td>150,641</td>
<td>9,992</td>
<td>786 (613)</td>
<td>3,950 (3%)</td>
</tr>
<tr>
<td>mailnesia</td>
<td>106,850</td>
<td>9,983</td>
<td>1,738 (686)</td>
<td>4,957 (5%)</td>
</tr>
<tr>
<td>mailfall</td>
<td>75,179</td>
<td>9,731</td>
<td>3,130 (288)</td>
<td>75,164 (100%)</td>
</tr>
<tr>
<td>mailsac</td>
<td>45,324</td>
<td>9,987</td>
<td>11,469 (8,019)</td>
<td>45,315 (100%)</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>2,332,544</strong></td>
<td><strong>56,589</strong></td>
<td><strong>452,220 (210,373)</strong></td>
<td><strong>1,290,073 (55%)</strong></td>
</tr>
</tbody>
</table>

Table 5.2: Statistics of the collected datasets.

48% of disposable email addresses received only one email, and 5% of popular addresses received more than 100 emails each.

Each email message is characterized by an email title, email body, receiver address (disposable email address), and sender address. As shown in Table 5.2, not all emails contain all the fields. 4 of the 7 disposable email services do not always keep the sender email addresses. Sometimes the disposable email services would intentionally or accidentally drop sender addresses. In addition, spam messages often omit the sender address in the first place. In total, there are 1,290,073 emails (55%) containing a sender address (with a total of 452,220 unique sender addresses). These sender addresses correspond to 210,373 unique sender domain names. From the email body, we extracted 13,396,757 URLs (1,031,580 unique URLs after removing URL parameters).

**Biases of the Dataset.** This dataset provides a rare opportunity to study disposable email services and email tracking. However, given the data collection method, the dataset inevitably suffers from biases. We want to clarify these biases upfront to provide a more accurate interpretation of the analysis results later. First, our dataset only covers the user-specified addresses but not the randomly-assigned addresses. Second, our data collection is complete with respect to the popular email addresses we monitored, but is incomplete with respect to all the available addresses. As such, any “volume” metrics can only serve as a lower bound. Third, we don’t claim the email dataset is a representative sample of a “personal inbox”. Intuitively, users (in theory) would use disposable email addresses differently relative to their personal email addresses. Instead, we argue the unique value of this dataset is that it covers a wide range of online services that act as the email senders. The data allows us to empirically study email tracking from the perspective of online services (instead of the perspective of email users). It has been extremely difficult (both technically and ethically) for researchers to access and analyze the email messages in users’ personal inboxes. Our dataset, obtained from public email gateways, allows us to take a first step measuring the email tracking ecosystem.
5.2.3 Ethical Considerations and IRB

We are aware of the sensitivity of the dataset and have taken active steps to ensure research ethics: (1) We worked closely with IRB to design the study. Our study was reviewed by IRB and received an exemption. (2) Our data collection methodology is designed following a prior research study on disposable SMS services [247]. Like previous researchers, we carefully have controlled the crawling rate to minimize the impact on the respective services. For example, we enforce a 1-second break between queries and explicitly use a single-thread crawler for each service. (3) All the messages sent to the gateways are publicly available to any Internet users. Users are typically informed that other users can also view the emails sent to these addresses. (4) We have spent extensive efforts on detecting and removing PII and personal emails from our dataset (details in §5.3.1). (5) After data collection, we made extra efforts to reach out to users and offer users the opportunity to opt out. More specifically, we send out an email to each of the disposable email addresses in our dataset, to inform users of our research activity. We explained the purpose of our research and offered the opportunity for users to withdraw their data. So far, we did not receive any data withdraw request. (6) Throughout our analysis, we did not attempt to analyze or access any individual accounts registered under the disposable email addresses. We also did not attempt to click on any URLs in the email body (except the automatically loaded tracking pixels). (7) The dataset is stored on a local server with strict access control. We keep the dataset strictly to ourselves.

Overall, we believe the analysis results will benefit the community with a deeper understanding of disposable email services and email tracking, and inform better security practices. We hope the results can also raise the awareness of the risks of sending sensitive information over public channels.

5.3 Analyzing Disposable Emails

In this section, we analyze the collected data to understand how disposable email services are used in practice. Before our analysis, we first detect and remove PII and the potential personal emails from the dataset. Then we classify emails into different types and infer their use cases. More specifically, we want to understand what types of online services with which users would register. Further, we seek to understand how likely it is for disposable email services to be used in sensitive tasks such as password resets.

5.3.1 Removing PII and Personal Emails

Removing PII. Since email messages sent to these gateways are public, we suspect careless users may accidentally reveal their PII. Thus, we apply well-established methods to detect and remove the sensitive PII from the email content [270]. Removing PII upfront
5.3. Analyzing Disposable Emails

Table 5.3: PII detection accuracy based on ground-truth, and the number of detected PII instances in our dataset.

<table>
<thead>
<tr>
<th>PII Type</th>
<th>#Email</th>
<th>#Inst.</th>
<th>F1</th>
<th>Precision</th>
<th>Recall</th>
<th># Detected in Our Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Credit</td>
<td>16</td>
<td>25</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1,399</td>
</tr>
<tr>
<td>SSN</td>
<td>13</td>
<td>15</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>926</td>
</tr>
<tr>
<td>EIN</td>
<td>16</td>
<td>29</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>701</td>
</tr>
<tr>
<td>Phone</td>
<td>20</td>
<td>50</td>
<td>0.99</td>
<td>0.98</td>
<td>1.00</td>
<td>726,138</td>
</tr>
<tr>
<td>VIN</td>
<td>15</td>
<td>19</td>
<td>0.97</td>
<td>1.00</td>
<td>0.95</td>
<td>43,438</td>
</tr>
</tbody>
</table>

allows us to analyze the dataset (including manual examination) without worrying about accidentally browsing sensitive user information. Here, we briefly introduce the high-level methodology and refer interested readers to [270] for details. The idea is to build a list of regular expressions for different PII. We first compile a ground-truth dataset to derive regular expressions and rules. Like [270], we also use the public Enron Email Dataset [48] which contains 500K emails. We focused on the most sensitive PIIs and labeled a small ground-truth set for credit card numbers, social security numbers (SSN), employer identification numbers (EIN), phone numbers, and vehicle identification numbers (VIN) as shown in Table 5.3. Then we build regular expressions for each PII type. For credit card numbers, we check the prefix for popular credit card issuers such as VISA, Mastercard, Discover and American Express, and we also use Luhn algorithm [202] to check the validity of a credit card number. As shown in Table 5.3, the regular expressions have good precision and recall.

We applied the regular expressions to our dataset and detected a large number of PIIs including 1,399 credit card numbers, 926 SSNs, 701 EINs, and 40K VINs and 700K phone numbers. All the detected PII are automatically blacked-out by the scripts. Note that the 700K phone numbers are not necessarily users’ personal phone numbers, but can be phone numbers of the email sending services. We take a conservative approach to blackout all the potential PII. The results indicate that people indeed use the disposable email services to communicate sensitive information.

Removing Personal Emails. We further remove potentially personal emails including replied emails and forwarded emails. We filter these emails based on “Re: ” and “Fwd: ” in the email titles. Although this step may not be complete, it helps to delete email conversations initiated by the users. In total, we filter out 30,955 such emails (1.33%). This again shows use of disposable email addresses for personal communications.

5.3.2 Categorizing Disposable Emails

Next, using the remaining data, we infer the common use cases of disposable email services by classifying email messages. First, we manually analyze a sample of emails to extract the
high-level categories of emails (ground-truth dataset). Second, we build a machine learning classifier and use it to classify the unlabeled emails. Third, we analyze the classification results to examine common usage cases.

**Manual Analysis and Email Clustering.** To assist the manual analysis, we first cluster similar email messages together. For efficiency considerations, we only consider the subject (or title) of the email message for the clustering. Since we don’t know the number of clusters in the dataset, we exclude clustering methods that require pre-defining the number of clusters (e.g., K-means). Instead, we use ISODATA algorithm [73] which groups data points based on a cut-off threshold of the similarity metric. We use Jaccard index to measure the keyword similarity of two email subjects. Given two email subjects, we extract all their keywords into two sets $w_i$ and $w_j$. Then we calculate their similarity as $\text{sim}(i, j) = \frac{|w_i \cap w_j|}{|w_i \cup w_j|}$.

We set the cut-off threshold as 0.2 to loosely group similar email titles together. In total, we obtain 91,306 clusters, most of which are small with less than 100 emails (98%). The cluster size distribution is highly skewed. The top 500 clusters cover 56.7% of the total email messages. A few large clusters (with over 1000 emails) typically represent spam campaigns. To make sure 0.2 is a reasonable threshold, we have tried even smaller thresholds to merge some of the clusters. For example, if we set the threshold to 0.1 and 0.01, we get 26,967 and 19,617 clusters respectively. However, manual examination shows that the emails in the same cluster no longer represent a meaningful group. We stick to 0.2 as the threshold. By manually examining 500+ clusters (prioritizing larger ones), we summarize 4 major types of emails.

- **Account Registration**: emails to confirm account registration in online services.
- **Password Reset**: emails that instruct the user to reset passwords for an online account.
- **Authentication**: emails that contain a one-time authentication code for login.
- **Spam**: all other unsolicited emails including newsletters, advertisements, notifications from online services, and phishing emails.

**Email Classification.** We need to further develop an email classifier because the clusters do not map well to each of the email categories. For example, a cluster may contain both spam emails and emails that are used to confirm account registration. Below, we build a machine learning classifier to classify emails into the four categories.

For classifier training, we manually labeled a ground-truth dataset of 5,362 emails which contains 346 account registration emails, 303 password reset emails, 349 authentication emails and 4,364 spam emails. Note that we have labeled more spam emails than other categories because our manual examination suggests that there are significantly more spam emails in the dataset. For each email, we combine the text in the email title and the email body, and
5.3. Analyzing Disposable Emails

apply RAKE (Rapid Automatic Keyword Extraction) [252] to extract a list of keywords. RAKE is a domain independent keyword extraction algorithm based on the frequency of word appearance and its co-occurrence with other words. In this way, less distinguishing words such as stopwords are automatically ignored. We use extracted keywords as features to build a multi-class SVM classifier. We have tested other algorithms such as Decision Tree and Random Forests. However, the SVM performed the best. We also tested word2vector [215] to build the feature vector, and its results are not as good as RAKE (omitted for brevity).

Through 5-fold cross-validation, we obtain a precision of 97.23% and a recall of 95.46%. This is already highly accurate for a multi-class classifier — as a baseline, a random classification over 4 classes would return an accuracy of 25%. We manually checked some of the classification errors, and found that a few account registration and authentication emails are labeled as spam due to “spammy” keywords (e.g., “purchase”).

Note that two types of emails are not applicable here. First, 58,291 (2.50%) of the emails do not have any text content. Second, 535,792 (22.97%) emails are not written in English. Since our classifier cannot analyze the text of these emails, they are not part of the classification results in Figure 5.1 (we still consider these emails in the later analysis of email tracking). To make sure our classification results are trustworthy, we randomly sampled 120 emails (30 per category) to examine manually. We only find 5 misclassified emails (4% error rate), which shows that the ground-truth accuracy transfers well onto the whole dataset.

5.3.3 Inferring Usage Cases

Next, we examine disposable email service usage. Recall that our dataset contains emails received by the disposable email addresses. Intuitively, after the users obtain the disposable email addresses, they will use the email addresses for certain online tasks (e.g., registering accounts), which will expose the addresses and attract incoming emails. By analyzing these incoming emails, we can infer at which services the user registered the accounts, and what the accounts are used for.
Table 5.4: Top 5 sender domains of registration emails, password reset emails and authentication emails.

<table>
<thead>
<tr>
<th>Rk.</th>
<th>sender domain</th>
<th># msg</th>
<th>category</th>
<th>sender domain</th>
<th># msg</th>
<th>category</th>
<th>sender domain</th>
<th># msg</th>
<th>category</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>facebookmail.com</td>
<td>2,076</td>
<td>Social Net</td>
<td>facebookmail.com</td>
<td>931</td>
<td>Social Net</td>
<td>frys.com</td>
<td>987</td>
<td>Shopping</td>
</tr>
<tr>
<td>2</td>
<td>gmail.com</td>
<td>1,015</td>
<td>Webmail</td>
<td>twitter.com</td>
<td>508</td>
<td>Social Net</td>
<td>paypal.com</td>
<td>622</td>
<td>Business</td>
</tr>
<tr>
<td>3</td>
<td>aol.com</td>
<td>928</td>
<td>Search</td>
<td>miniclip.com</td>
<td>415</td>
<td>Games</td>
<td>ssl.com</td>
<td>418</td>
<td>IT</td>
</tr>
<tr>
<td>4</td>
<td>avendata.com</td>
<td>733</td>
<td>Business</td>
<td>retailio.in</td>
<td>223</td>
<td>Business</td>
<td>id.com</td>
<td>163</td>
<td>Business</td>
</tr>
<tr>
<td>5</td>
<td>axway.com</td>
<td>720</td>
<td>Education</td>
<td>gmail.com</td>
<td>145</td>
<td>Webmail</td>
<td>facebookmail.com</td>
<td>161</td>
<td>Social Net</td>
</tr>
</tbody>
</table>

Types of Emails. As shown in Figure 5.1, while spam emails take the majority, there is a non-trivial number of emails that are related to account management in various online services. In total, there are 89,329 emails involved with account registration, password resets or sending authentication codes. These emails are sent from 168,848 unique web domains. We refer these 3 types of emails as account management emails. Account management emails are indicators of previous interactions between the user and the email sending domain. They are explicit evidence that users have used the disposable email addresses to register accounts in the web services.

Breakdown of Spam Emails. The spam emails take a large portion of our dataset (1,612,361 emails, 94%), which deserve a more detailed break-down. Some of the spam messages also indicate previous interactions between a user and the email sender. For example, if a user has registered an account or RSS at an online service (e.g. Facebook), this service may periodically send “social media updates”, “promotions”, or “notifications” to the disposable email address. We call them notification spam. Such notification messages almost always include an unsubscribe link at the bottom of the email to allow users to opt out. As such, we use this feature to scan the spam messages and find 749,602 notification messages (counting for 46.5% of the spam messages).

The rest of unsolicited spam messages may come from malicious parties, representing malware or phishing campaigns. To identify the malicious ones, we extract all the clickable URLs from the email content, and run them against the VirusTotal blacklists (which contains over 60 blacklists maintained by different security vendors [71, 247]), and the eCrimeX blacklist (a phishing blacklist maintained by the Anti Phishing Work Group). In total, we identify 84,574 malicious spam emails (5.2%) that contain at least one blacklisted URL.

Finally, we apply the same ISODATA clustering algorithm to the rest of the spam emails (which count for 48.3%) to identify spam campaigns. We find 19,314 clusters and the top 500 clusters account for 75.6% of the spam emails. Manual examination shows that the top clusters indeed represent large spam campaigns, most of which are pornography and pharmaceutical spam.

Categories of Email Senders. To understand what types of online services users interact with, we further examine the “categories” of email sender domains. The “categories” are provided by VirusTotal. Table 5.5 shows the top 10 categories for spam emails and
## 5.3. Analyzing Disposable Emails

<table>
<thead>
<tr>
<th>Rk.</th>
<th>Account Management Email</th>
<th>Spam Email</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Category</td>
<td># Msg (domain)</td>
</tr>
<tr>
<td>1</td>
<td>Business</td>
<td>12,699 (2,079)</td>
</tr>
<tr>
<td>2</td>
<td>IT</td>
<td>6,759 (1,228)</td>
</tr>
<tr>
<td>3</td>
<td>Software</td>
<td>5,481 (571)</td>
</tr>
<tr>
<td>4</td>
<td>Social Net</td>
<td>5,362 (149)</td>
</tr>
<tr>
<td>5</td>
<td>Marketing</td>
<td>5,320 (430)</td>
</tr>
<tr>
<td>6</td>
<td>Shopping</td>
<td>3,307 (370)</td>
</tr>
<tr>
<td>7</td>
<td>Education</td>
<td>2,946 (673)</td>
</tr>
<tr>
<td>8</td>
<td>Search</td>
<td>2,154 (74)</td>
</tr>
<tr>
<td>9</td>
<td>Finance</td>
<td>2,017 (302)</td>
</tr>
<tr>
<td>10</td>
<td>Webmail</td>
<td>1,575 (46)</td>
</tr>
</tbody>
</table>

Table 5.5: Top 10 categories of the email sender domains for spam and account management emails.

We have two main observations. First, the emails are sent from a very broad range of domain categories. This suggests that users have used the disposable email addresses to register accounts in all different types of websites. There are in total 121 different categories, and the top-10 categories only cover 51.01% of account management emails and 58.25% of spam emails, which confirms the high diversity of usage. Second, we observe that disposable email addresses are often used to register potentially sensitive accounts. Categories such as “online social networks”, “finance”, “shopping” have made the top-10 for account management emails. This could introduce risks if a user accidentally left PII or credit card information in the registered account. Accounts registered under disposable email addresses are easily hijackable. Any other users can take over the registered accounts by sending a password-reset link to the disposable email address, which will be publicly accessible. Given the 14,000+ password-reset emails in our dataset, it is possible that malicious parties are already performing hijacking.

**Case Studies: Common Usage.** Next, we use specific examples to illustrate the common usage cases. Table 5.4 lists the top 5 email sending domains for registration, password reset and authentication emails. We show users use disposable email addresses to register accounts in gaming and social network services in order to enjoy the online services without giving away real email addresses. For example, facebookmail.com appears in the top-5 of all three types of emails. twitter and miniclip (for gaming) also fall into the same category. It is possible that some accounts are fake accounts registered by spammers [293]. Since we decided not to back-track (or login into) any individual user’s account for ethical considerations, we cannot systematically differentiate them. Previous research on anonymous community (e.g., 4chan, Reddit) show that users prefer anonymized identifiers when posting sensitive or controversial content [206, 283]. We suspect normal users may use the
disposable email address to create such social media accounts for similar purposes. PayPal accounts have additional risks. If a user accidentally binds a real credit card to the account, it means any other users may take over the PayPal account by resetting the password.

Another common use case is to obtain free goods. For example, users often need to register an email address to obtain demos or documents from software solutions and educational services, e.g., axway.com, avendata.com, retailio.in, and ssl.com. Users can also obtain a discount code from shopping services (e.g., frys.com). Another common case (not in the top-5) is to use the disposable email address to register for free WiFi in airports and hotels. Finally, we observe cases (not in the top 5) where users try to preserve anonymity: For example, people used disposable email addresses to file anonymous complaints to the United States Senate (86 emails).

Note that gmail.com is special: it turns out that many small businesses cannot afford their own email domains and directly use Gmail (e.g., pizza@gmail.com). Thus, The domain gmail.com does not represent Gmail, but is a collection of small businesses. aol.com has a similar situation.

Case Studies: Risky Usage. We observe other cases that may involve risks. These cases may be not as common as those shown in Table 5.4, but if their accounts are hijacked (through the public disposable email addresses), the real-world consequences are more serious. For example, there are 4,000+ emails from healthcare.gov, the website of the Affordable Care Act. It is likely that people have used disposable email addresses to register their healthcare accounts where each account carries sensitive information about the user.

Similarly, there are emails from mypersmail.af.mil (Air Force Service Center), suggesting that people have used disposable email address to register Air Force personnel accounts. The registration is open to civilian employees who must use their SSN and date of birth for the registration [2]. A password reset option is also available on the website.

In addition, more than 32,990 emails are used to receive scanned documents from PDF scanning apps (e.g., Tiny Scanner). It is possible for an attacker to obtain all the scanned documents by hijacking these disposable email addresses.

Finally, there are over 1000 emails from digital currency or digital wallet services such as buyabitcoin.com.au and thebillioncoin.info. While most emails are related to account registrations, some are related to bitcoin purchase confirmations (e.g., receipts). If these accounts hold bitcoins, anyone has a chance to steal them.

5.3.4 Summary

We show that disposable email services are primarily used to register online accounts. While most of the incoming emails are spam and notifications (94%), we did find a large number of emails (89,000+) that are related to account registration, password reset, and login au-
5.4. Email Tracking Measurements

There is a strong evidence that users use disposable email services for sensitive tasks. We find 1000+ credit card numbers and 926 SSNs accidentally revealed in the emails and 30K replied and forwarded emails that indicate a personal usage. More importantly, accounts registered with disposable email addresses can be easily hijacked through a password reset.

5.4 Email Tracking Measurements

Next, we use the large-scale email dataset to analyze email tracking in the wild. We seek to answer three key questions. First, what types of tracking techniques do trackers use in practice, and what is the nature of the data leaked through tracking. Second, how prevalent is third-party tracking among different types of online services? Third, who are the top trackers in the tracking ecosystem and how dominant are they? In the following, we first describe the threat model and our method to detect third-party tracking, and then present the measurement results.

5.4.1 Threat Model

By embedding a small image in the email body, the email sender or third-parties can know whether the email has been opened by the receiver. When an email is opened, the tracking pixel will be automatically loaded from a remote server via HTTP/HTTPS (which does not require any user actions). Based on the request, the remote server will know who (based on the email address or other identifiers) opened the email at what location (based on IP) and what time (timestamp) using what device (“User-Agent”). The privacy leakage is more serious when the remote server is a third-party.

Email tracking works only if the user’s email client accepts HTML-based email content, which is true for most modern email clients. However, careful users may use ad-blockers to block tracking pixels [120]. In this chapter, we make no assumption about a user’s email client, and only focus on the tracking content in the email body. Note that JavaScript is not relevant to email tracking since JavaScript will not be automatically executed [27]. Alternatively, email tracking can be done through querying font files. We did not find any font-based tracking in our dataset and omit it from the threat model.

5.4.2 Tracking Detection Method

Given an email, we design a method to determine if the email contains tracking pixels. First, we survey popular email tracking services (selected through Google searching) to examine how they implement the tracking pixels. After analyzing Yesware, Contact Monkey,
Chapter 5. Email Pixel Tracking through Disposable Email Services

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Tracking Stats</th>
<th>Tracking Party*</th>
<th>Tracking Method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total</td>
<td>1st-party</td>
<td>3rd-party</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Plaintext</td>
<td>Obfuscated</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Invis. HTML</td>
<td>Invis. remote</td>
</tr>
<tr>
<td># Image URLs</td>
<td>3,887,658</td>
<td>509,119</td>
<td>200,682</td>
</tr>
<tr>
<td></td>
<td>(1.222,961 (31.5%))</td>
<td>170,223</td>
<td>200,247</td>
</tr>
<tr>
<td></td>
<td></td>
<td>548,166</td>
<td>537,266</td>
</tr>
<tr>
<td># Email Messages</td>
<td>2,332,544</td>
<td>264,501</td>
<td>35,702</td>
</tr>
<tr>
<td></td>
<td>(573,244 (24.6%))</td>
<td>149,303</td>
<td>29,445</td>
</tr>
<tr>
<td></td>
<td></td>
<td>473,723</td>
<td>124,900</td>
</tr>
<tr>
<td># Sender Domains</td>
<td>210,373</td>
<td>5,403</td>
<td>1,478</td>
</tr>
<tr>
<td></td>
<td>(11,688 (5.5%))</td>
<td>7,398</td>
<td>9,149</td>
</tr>
<tr>
<td></td>
<td></td>
<td>9,935</td>
<td>1,802</td>
</tr>
<tr>
<td># Tracker Domains</td>
<td>N/A</td>
<td>13,563</td>
<td>5,381</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2,302</td>
<td>9,149</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2,403</td>
<td>1,802</td>
</tr>
<tr>
<td></td>
<td></td>
<td>984</td>
<td>9,935</td>
</tr>
<tr>
<td></td>
<td></td>
<td>9,935</td>
<td>1,802</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1,802</td>
<td>9,935</td>
</tr>
</tbody>
</table>

Table 5.6: Email tracking detection results. *Tracking party is based on 1.29 million emails that have a sender address.

Figure 5.2: Distribution of the HTML image size.

Figure 5.3: The HTML image size of invisible remote pixels.

Figure 5.4: # of tracking URLs under different tracking methods.

Mailtrack, Bananatag, Streak, MailTracker, The Top Inbox, and Hub Spot, we observe two common characteristics. First, all 8 services embed small or transparent HTML image tags that are not visible to users (to remain stealthy). Second, the image URLs often contain some form of user identifiers (either the receiver's email address or IDs created by the tracking services). This is because the tracker wants to know “who” opened the email. Next, we design a detection method based on these observations.

Steps to Detect Pixel Tracking. Given an email, we first extract all the HTML image tags and corresponding URLs. Here, we focus on tracking URLs that notify the tracker about the user identity. We filter out links that do not contain any parameters. Then for each image URL, we follow the four steps below to detect email tracking.

- **Step 1: Plaintext Tracking Pixel**: if the link’s parameters contain the receiver’s email address in plaintext, then the image is a tracking pixel.

- **Step 2: Obfuscated Tracking Pixel**: if the link’s parameters contain the “obfuscated version” of the receiver’s email address, then the image is a tracking pixel. We apply 31 hash/encoding functions on the receiver email address to look for a match (see Appendix). We also test two-layer obfuscations by exhaustively applying two-function combinations, e.g., MD5(SHA1()). In total, we examine 992 obfuscated strings for each address. We didn’t consider salted obfuscation here due to the extremely high testing complexity.

2Image URLs without parameters will still reveal the user’s IP but are not necessarily for tracking
5.4. Email Tracking Measurements

- **Step 3: Invisible HTML Pixel**: we check if the image is trying to hide based on the HTML height and width attributes. We consider the image as a tracking pixel if both the height and width are below a threshold $t$ or the HTML tag is set to be “hidden” or “invisible”.

- **Step 4: Invisible Remote Pixel**: trackers may intentionally set a large height or width in HTML to avoid detection. If the HTML height or width is above $t$, we use a web crawler to fetch the actual image from the remote server. If the actual image size is below $t$, regardless the HTML attributes, we regard it as a tracking pixel.

Step-1 and step-2 are adapted from the method described in [120]. We explicitly look for parameters in the image URL that leak the receiver’s email address. However, it is still possible that trackers use an obfuscation method that is not listed in Table B.1 (e.g., keyed-hash). More importantly, the tracker can use a random string as the identifier and keep the mapping in the back-end. As such, we introduce step 3 and step 4 as a complementary way to capture the tracking behavior that cannot be detected by [120].

To set the threshold $t$ for tracking pixels, we plot Figure 5.2 to show the image size distribution in our dataset. Image size is defined as the larger value between the height and width. As shown in Figure 5.2, there is a clear peak where the image size is 1 (1.1 million images). There are also 60K images of a “zero” size. To be conservative, we set the threshold $t = 1$. Our method is still not perfect, since we might miss trackers that use bigger tracking images. The detection result is only a lower-bound of all possible tracking.

**Alternative Tracking Methods.** In addition to the methods above, we have tested other alternative methods, which did not return positive results in our pilot test. For completeness, we briefly discuss them too. First, other than URL parameters, trackers use subdomain names to carry the user identifiers. For example, a tracker (e.g., tracker.com) may register many subdomains, and use each subdomain to represent a user (e.g., u1.tracker.com, u2.tracker.com). To look for such trackers, we sort the domain names of image URLs based on their number of subdomains. We only find 3 domain names (list-manage.com, sendgrid.com and emltrk.com) that have more than 1000 subdomains. However, we find that they are not using subdomain names as user identifiers. Instead, each subdomain is assigned to represent a “customer” that adopted their tracking services. For example, a tracking URL office-artist.us12.list-manage.com is used by online service office-artist.com to track their users. We have examined all the tracking domains with over 50 subdomains and did not find any subdomain-based tracking.

A limitation of step-1 and step-2 is that they cannot capture trackers that use a random string as the identifier. An alternative approach is cluster image URLs that follow the same templates. Then the differences in the URLs are likely to be the unique user identifiers. However, our pilot test shows that the majority of the differences in image URLs are indeed personalized content, but the personalized content is not for tracking. For example, online
services often send product recommendations using the same template but use different “ProductIDs” in the image URLs. This approach easily introduces false positives.

**Third-party Tracking.** To differentiate first-party and third-party tracking, we match the domain name of the *email sender* and that of the *image URL*. Since we use domain name to perform the matching, all the “subdomains” belong to the same party. For example, mail.A.com and image.A.com match with each other since they share the same domain name. If the email sender’s domain name is different from that of the image tracking URL, we then check their WHOIS record to make sure the two domains are not owned by the same organization. We regard the tracking as a third-party tracking if the two domain names belong to different organizations.

### 5.5 Measurement Results

We apply our detection method to the 2.3 million emails, and the results are summarized in Table 5.6. In total, we extracted 3.9 million unique image URLs and 1.2 million of them (31.5%) are identified as tracking links. These tracking links are embedded in 573K emails (24.6%). Out of the 210K email sender domains, we find that 11.6K of them (5.5%) have embedded the tracking pixels in their emails. In total, we identify 13,563 unique tracker domains. In the following, we first characterize different email tracking techniques and the “hidden trackers”. Then we focus on third-party tracking and identify the top trackers. Finally, we analyze how different online services perform tracking.

#### 5.5.1 Email Tracking Techniques

As shown in Table 5.6, there is almost an equal number of tracking URLs that send plaintext user identifiers (200,682) and those that send obfuscated identifiers (200,247). For the obfuscated tracking, we find 12 obfuscated methods are used by trackers (out of 992 obfuscations tested). As shown in Table 5.7, MD5 is applied in the vast majority of these tracking URLs.
5.5. Measurement Results

Figure 5.5: Different tracking methods of first-party and third-party trackers.

Figure 5.6: # of third-party trackers per sender.

Figure 5.7: # of sender domains associated to each tracker.

(91.7%) followed by Base64 (4.9%). We did find cases where the obfuscation functions are applied more than once but these cases are rare (<0.5%). This observation is consistent with the previous smaller-scale study [120].

There are even more tracking links that use invisible pixels. We find 548,166 invisible HTML pixels where the HTML size attributes are 1×1 or smaller or the image tags are set to be “hidden”. Meanwhile, we find 537,266 additional invisible remote pixels which falsely claim their HTML size attributes even though the actual image is 1×1. By analyzing the HTML attributes of the invisible remote pixels, we find that 20% of them did not specify the size attributes. For the remaining images that specified the size, Figure 5.3 shows the size distribution. These pixels declare much larger image sizes in HTML (possibly to avoid detection) while the actual image is only 1×1 (invisible to users).

Figure 5.4 shows the overlaps of the tracking URLs detected by different methods. We find 17K (8.6%) the plaintext tracking URLs are also using invisible HTML pixels; 114K (56.8%) plaintext tracking URLs are using invisible remote pixels. This suggests that trackers prefer stealthier methods when sending plaintext identifiers. For obfuscated tracking URLs, although the “remote” invisible pixels are still preferred (86K, 42.7%), the ratio is more balanced compared to the usage of HTML pixels (47K, 23.3%). When the parameters are obfuscated, the trackers are likely to put in less effort towards hiding their tracking pixels.

Hidden Trackers. Through our analysis, we find hidden trackers when we try to fetch the tracking pixels from the remote servers. More specifically, when we request the images, the request will be first sent to the “direct tracker” (based on the image URL) and then redirected to the “hidden trackers”. The hidden trackers are not directly visible in the email body and can only be reached through HTTP/HTTPS redirections. In this way, user identifiers are not only leaked to the direct tracker but also to the hidden trackers in real time. Intuitively, hidden trackers are less likely to be blacklisted (by adblockers) since they do not directly appear in the HTML. To capture hidden trackers, we crawled all of the 1,222,961 tracking URLs. We find that a large number of the tracking URLs have redirections (616,535, 50.4%). In total, we obtain 2,825 unique hidden tracker domains. Table 5.8 shows the top 10 hidden trackers (ranked by the number of the direct trackers that redirect traffic
to them).

Hidden trackers may also act as direct trackers in certain emails. We find that 2,607 hidden trackers have once appeared to be direct trackers in our dataset. In total, hidden trackers are associated with 112,068 emails and 2260 sender domains (19.3% of sender domains that adopted tracking). Interestingly, many first-party tracking links also share the user information with hidden trackers in real-time. More specifically, there are 9,553 emails (220 sender domains) that share user identifiers to a hidden tracker while performing first-party tracking.

### 5.5.2 Third-party Tracking

Next, we focus on third-party tracking and identify the top trackers. This analysis is only applicable to emails that contain a sender address (1.2 million emails).

**Overall Statistics.** Third-party tracking is highly prevalent. As shown in Table 5.6, there are 149k emails with third-party tracking. Interestingly, there are more sender domains with third-party tracking (7,398) than those with first-party tracking (5,403). In total, we identify 2,302 third-party trackers.
5.5. Measurement Results

<table>
<thead>
<tr>
<th>Rk.</th>
<th>Top Trackers</th>
<th>Type</th>
<th># Senders</th>
<th># Emails</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>list-manage.com</td>
<td>o</td>
<td>1,367</td>
<td>19,564</td>
</tr>
<tr>
<td>2</td>
<td>sendgrid.net</td>
<td>o</td>
<td>849</td>
<td>10,416</td>
</tr>
<tr>
<td>3</td>
<td>returnpath.net</td>
<td>o</td>
<td>345</td>
<td>12,784</td>
</tr>
<tr>
<td>4</td>
<td>rs6.net</td>
<td>o</td>
<td>292</td>
<td>3,274</td>
</tr>
<tr>
<td>5</td>
<td>emltrk.com</td>
<td>o</td>
<td>226</td>
<td>3,328</td>
</tr>
<tr>
<td>6</td>
<td>google-analytics.com</td>
<td>o</td>
<td>225</td>
<td>5,174</td>
</tr>
<tr>
<td>7</td>
<td>doubleclick.net</td>
<td>●</td>
<td>208</td>
<td>12,968</td>
</tr>
<tr>
<td>8</td>
<td>hubspot.com</td>
<td>●</td>
<td>192</td>
<td>874</td>
</tr>
<tr>
<td>9</td>
<td>eloqua.com</td>
<td>●</td>
<td>150</td>
<td>1,981</td>
</tr>
<tr>
<td>10</td>
<td>rlcdn.com</td>
<td>●</td>
<td>133</td>
<td>7,117</td>
</tr>
<tr>
<td></td>
<td><strong>Subtotal</strong></td>
<td></td>
<td><strong>3,715 (31.8%)</strong></td>
<td><strong>68,914 (12.0%)</strong></td>
</tr>
</tbody>
</table>

Table 5.10: Top third-party trackers across the full dataset. “●” means the tracker is also a hidden tracker. “o” means the tracker is not a hidden tracker.

Figure 5.5 breaks down the tracking methods used by first- and third-party trackers. To make sure different tracking methods don’t overlap, we present plaintext tracking and obfuscated tracking, and regard the rest of the invisible pixel tracking as “other tracking”. Figure 5.5 shows that third-party trackers are less likely to collect the user email address as the identifier.

Figure 5.6 shows the number of third-party trackers used by each sender domain (corresponding to an online service). We find that the vast majority (83%) of online services use a single third-party tracker. About 17% of online services have multiple third-party trackers, sharing user information with multiple-parties at the same time. The extreme case is amazones.com which uses 61 third-party trackers.

**Top Trackers.** From the third-party tracker’s perspective, Figure 5.7 shows that only a small number of trackers are used broadly by different online services. To analyze the top trackers, we present Table 5.9 to list top third-party trackers for each tracking method. We rank the trackers based on the number of online services that use them. A popular tracker should be used by many online services. For reference, we also show the number of emails associated with each tracker.

We observe that top trackers under different tracking methods rarely overlap with each other. This indicates that a tracker usually sticks to a specific tracking method. The most dominating trackers per category are mczany.com (plaintext tracking), alcmpn.com (obfuscated tracking), list-manage.com (invisible HTML), and hubspot.com (invisible remote). Noticeably, under the “stealthy” remote tracking, we also find that google-analytics.com and doubleclick.net make the top 10, which are Google’s trackers that have dominated web tracking [60, 194, 265].

Table 5.10 shows the top trackers across the full dataset, including all the hidden trackers. We
show that the top 10 trackers collectively cover 33.5% of online services, and are responsible for 12% of the tracking emails. Although top trackers are taking a big share of the market, they are not as dominating as the top tracker (i.e. Google) in web tracking. For example, previous measurements show that Google can track users across nearly 80% of the top 1 million sites [199]. Clearly, in the email tracking market, Google is not yet as dominating as it is in the web tracking.

5.5.3 Tracking by Online Services

Finally, we analyze different online services and seek to understand whether the popularity of online services and the service type would correlate to different tracking behaviors.

Popular vs. non-Popular Online Services. We first examine how tracking correlates with the popularity of online services. We reference Alexa’s top 1 million domains for the ranking [4]. Note that Alexa’s ranking is primarily applied to the web domain instead of
the email domain. Accordingly, we check the MX record of Alexa top 1 million domains to perform the match. We find that out of the 210,373 sender domains, 18,461 domains are within Alexa top 1 million, and 2,052 are within the Alexa top 10K. For our analysis, we treat the Alexa top 10K as the popular domains, and the rest as non-popular domains. In total, the small portion of popular domains (0.98%) contributed 15.9% of the total emails.

Figure 5.8 shows that tracking is much more prevalent among popular domains. About 50% of popular domains adopted tracking in their emails. As a comparison, less than 10% of non-popular domains have adopted email tracking. Regarding different tracking methods, plaintext tracking and obfuscated tracking are not as prevalent as invisible pixel tracking, which is true for both popular and non-popular domains. Figure 5.9 shows that popular domains are slightly more likely to have first-party tracking than third-party tracking. Figure 5.10 shows that popular domains are more likely to use tracking methods that are harder to detect. More specifically, we focus on two types of stealthy tracking including: invisible remote pixels (where the HTML tags falsely claim the image size) and hidden trackers (trackers hide behind redirection). We observe a big difference: about 12% – 16% of popular domains have used stealthy tracking and only 1% of non-popular domains use such tracking methods.

Type of Online Services. In Figure 5.11, we focus on the top 10 categories of sender domains and analyze the ratio of them that adopted email tracking. Not too surprisingly, “marketing” services have the highest ratio of tracking. In fact, many marketing services themselves are email tracking services (first-party tracking). Popular tracking domains also include shopping websites and information technology websites.

5.6 Discussion

Risk Mitigation for Disposable Email Addresses. Our study reveals risky use cases of disposable email services. The root source of risk is the public nature of the disposable email inboxes. Randomly-assigned addresses cannot fully mitigate this problem since multiple users can still access the same address at the same time (see §5.2.1). One possible countermeasure is to implement sandbox using cookies. For example, if a current user is using the inbox, then other users who do not possess the same cookie cannot access the same inbox. The inbox will become available again once the current user closes her session. If the disposable email service does not implement sandbox, we believe it is necessary for the service to clearly inform users about the public nature of the inbox. In addition, it is also important for the service to clearly communicate the email expiration time to users. Our results show that two disposable email services host the emails much longer than what they promised (e.g., 30 days of delay).

Users of disposable email services should proactively delete their emails whenever possible.
More importantly, users should avoid revealing their PII in both the temporary inbox and in the accounts they registered through the disposable email address. Due to the public nature of the disposable email services, accounts registered with disposable email addresses can be easily hijacked through a password reset. A future direction is to understand user perceptions towards the benefits and risks of using disposable email services and identify the potential misunderstandings with respect to their security.

**Email Tracking and Countermeasures.** The most straightforward way to prevent email tracking is to stop rendering emails in HTML (i.e., plaintext email) or block all the outgoing requests that are not initiated by user clicks. The drawback, however, is a degradation of user experience since the images in the email (if they are not embedded) cannot be displayed. To address this problem, Gmail has a special design where the Gmail server fetches all the images on behalf of the users. In this way, the tracker cannot collect users’ IP addresses. However, the tracker can still obtain the following information: (1) the user indeed opens the email; (2) the time of email opening; and (3) the user’s identifier (if the identifier is a parameter of the tracking URL).

A more promising way is to perform targeted HTML filtering to remove tracking related image tags. Since most of tracking pixels are invisible, removing them would not hurt the user experience. This is very similar to ad-blocking where the ad-blocker construct filtering rules to detect and remove ads on websites. In addition to static HTML analysis, we believe dynamic analysis is necessary since (1) trackers may falsely claim the HTML size attributes, and (2) the real trackers may hide behind the redirection.

**Email Tracking Notification.** For the sake of transparency, it is necessary to inform users when tracking is detected. Today, many websites are required (e.g., by EU Privacy Directive) to display a notice to inform users when cookies are used for web tracking. More recently, EU’s new GDPR policy forbids online services from tracking users with emails without unambiguous consent. However, there is no such privacy policy in the U.S.. While legislation may take a long time, a more immediate solution is to rely on email services or email clients to notify users.

**A Comparison with Previous Research.** The most related work to ours is a recent study that analyzed emails tracking of 902 websites (12,618 emails) [120]. In this work, we collect a dataset that is larger by orders of magnitude. Some of our results confirm the observations of the small-scale study. For example, we show that obfuscation is widely used to encode user identifiers for tracking and MD5 is the most commonly used method, both of which are consistent with [120]. Interestingly, Some of our results are different, in particular, the top third-party trackers (Table 5.9). For example, doubleclick.net, which was ranked 1st by [120], is only ranked 7th based on unique sender domains (ranked 2nd based on email volume) in our dataset. list-manage.com was ranked 10th by [120] but came to the top in our analysis. There are a couple reasons that may contribute to the differences. First, the previous work collected a small email dataset from 902 sender domains, while we
collected emails from 210,000+ sender domains. Second, the previous study collected data from “Shopping” and “News” categories, while our dataset covers more than 100 website categories. Third, previous work only considered tracking URLs that contain an explicit user identifier (i.e., email address), while we cover more tracking methods (e.g., invisible or remote pixels).

Limitations The first limitation is that our analysis only covers disposable email services with user-specified addresses (UA). This is mainly due to the difficulty to obtain data from randomly-assigned addresses (RA). Here, we use the small dataset collected from RA services (§5.2.1) to provide some contexts. Recall the dataset contains 1,431 messages from 5 RA services. After removing personal and non-English emails, we apply our classifier to the rest 1142 emails. We find that randomly-assigned addresses also contain account management emails, including 134 registration emails (11.7%), 44 password reset emails (3.9%), and 32 authentication emails (2.8%). We also notice that the spam email ratio is lower in RA services (81.6%) than that of UA services (94%). Intuitively, spammers often blindly send spam emails to addresses with popular usernames.

The second limitation is that our dataset is not representative with respect to a normal user inbox. Our measurement results cannot be used to assess email tracking at a per-user level. Instead, the main advantage of the dataset is that it contains emails sent by a large number of online services (including the top-ranked websites). This allows us to analyze email tracking from the perspective of online services (200K domains across 121 categories). For future work, we can evaluate the user-level tracking through user studies.

Third, for ethical considerations, we decided not to manually analyze the PII or back-track the accounts registered with the disposable addresses. This has limited our ability to answer some of the questions. For example, in §5.3.1, we did not manually confirm the validity of detected PII, assuming the training accuracy transfers well to the testing. In §5.3.3, it is possible that spammers would use the email addresses to register fake accounts in online services, but we cannot confirm. Similarly, for the password reset emails, it is possible that the emails were triggered by malicious parties who were trying to login other people’s accounts, or by the real owners of the accounts who forgot the password.

Fourth, our email tracking detection is still incomplete. Theoretically, it is possible for a tracker to use subdomain names (instead of URL parameters) to identify individual users, or use font links (instead of image links). However, we did not find such cases in our dataset. In addition, our current method cannot detect tracking URLs that use both large tracking images and random strings as user identifiers.

5.7 Related Work

Web Tracking and Email Tracking. Web tracking has been extensively studied by
researchers in the past decade [87]. Researchers have analyzed third-party web tracking across different websites [194] and countries [157]. Consistently, different studies have shown that Google is the top tracker on the web [210, 250] where 80% of Alexa top 1 million websites have Google-owned trackers [199]. Web tracking has turned into a cat-and-mouse game. Researchers have studies various tracking techniques such as flash cookies [72, 262], canvas fingerprinting, evercookies, and cookie syncing [60, 118]. While adblockers help to reduce tracking, anti-adblockers are also increasingly sophisticated [158, 218, 225, 324].

Disposable Accounts and Phone Verified Accounts. Previous work has studied disposable SMS services where public phone numbers are offered to users for a temporary usage [247]. Researchers also studied the security risks of man-in-the-middle attack [137], and use the collected messages to investigate SMS spam [163, 220]. A recent work shows that “retired” addresses from popular email services can be re-registered to hijack existing accounts [140]. Other researchers looked in how disposable SMS are used to create phone-verified fake accounts in online services [272].

PII Leakage and Email Hijacking. Previous works have examined PII leakage under various channels [179, 180] such as mobile network traffic [249, 282], website contact forms [266], and cross-device tracking [86]. Our work differs from previous works with a focus on PII leakage during email tracking.

5.8 Conclusion

In this chapter, we perform a first measurement study on disposable email services. We collect a large dataset from 7 popular disposable email services (2.3 million emails sent by 210K domains), and provide new understandings of what disposable email services are used for and the potential risks of usage. In addition, we use the collected email dataset to empirically analyze email tracking activities. Our results provide new insights into the prevalence of tracking at different online services and the evasive tracking methods used of trackers. The results are valuable for developing more effective anti-tracking tools.
Chapter 6

Elite PhishingDomains in the Wild

6.1 Acknowledgement

The leading author of this work is Dr. Ke Tian. I am the third author of this work. Dr. Ke Tian led this project and conducted the most experiments. I helped by co-writing the squatting detection module and helped writing data crawlers to collect web page data. After the data collection, we worked together to label our data set for the training of the phishing classifier.

6.2 Introduction

Today, phishing attacks are increasingly used to exploit human weaknesses to penetrate critical networks. A recent report shows that 71% of targeted attacks began with a spear phishing \[56\], which is one of the leading causes of the massive data breaches \[53\]. By luring the targeted users to give away critical information (\textit{e.g.}, passwords), attackers may hijack personal accounts or use the obtained information to facilitate more serious attacks (\textit{e.g.}, breaching a company’s internal network through an employee’s credential) \[51\].

Phishing webpages, as the landing pages for phishing messages \[115, 184, 248\], are constantly involving to \textit{deceive users} and \textit{evade detection}. Sophisticated phishing pages are constructed to impersonate the webpages of banks, government agencies, and even the internal systems of major companies \[103\]. In addition, phishing pages can also impersonate the domain names of trusted entities via domain squatting techniques \[147, 172, 217\]. For example, an attacker may register a domain that looks like facebook.com using an internationalized domain name to deceive users, as shown in Figure 6.1. While anecdote evidence suggests such “\textit{elite}” phishing pages exist, there is still a lack of in-depth understandings of how the phishing pages are constructed and used in practice.

In this chapter, we describe our efforts in searching and detecting \textit{squatting phishing domains} where the attackers apply impersonation techniques to both the web content and the web domain. Our goals are threefold. First, we seek to develop a systematic method to search and detect squatting phishing domains in the wild. Second, we aim to empirically examine the \textit{impersonation} techniques used by the attackers to deceive users. Third, we want to char-
Chapter 6. Elite Phishing Domains in the Wild

Figure 6.1: An example of the internationalized domain name \texttt{xn--facbook-ts4c.com} (homograph), which is displayed as \texttt{facebook.com} in the address bar.

characterize the \textit{evasion} techniques used by the squatting phishing pages and their effectiveness to avoid detection.

To these ends, we design a novel measurement system \texttt{SquatPhi} to search and detect squatting phishing domains. We start by detecting a large number of “squatting” domains that are likely to impersonate popular brands. Then, we build a distributed crawler to collect the webpages and screenshots for the squatting domains. Finally, we build a machine learning classifier to identify squatting phishing pages. A key novelty is that our classifier is built based on a careful measurement of the evasion techniques used by real-world phishing pages. These evasion techniques are likely to render existing detection methods ineffective. Below, we describe each step and the discuss our key findings.

\textbf{Squatting Domain Detection.} We focus on 702 highly popular online services (brands) and search for squatting domains that are likely to impersonate them (\textit{e.g.}, Facebook, PayPal). We apply five different squatting techniques \cite{172, 224, 271} to generate candidate domains, including typo squatting, bits squatting, homograph squatting, combo squatting, and wrongTLD squatting. By analyzing over 224 million DNS records, we identified 657,663 squatting domains, and crawled both the web version and mobile version of their webpages (1.3 million pages) for 4 snapshots over a month.

\textbf{A Novel Phishing Classifier.} To detect squatting phishing pages among a large number of squatting domains, we develop a novel machine learning classifier. Based on a ground-truth set of 4004 user-reported phishing pages (from PhishTank \cite{33}), we characterize common evasion techniques, and develop new features as countermeasures. Particularly, we observe that evasion techniques (\textit{e.g.}, code obfuscation, string obfuscation, and layout obfuscation) often hide phishing related text in the source code or change the layout of the phishing pages. To this end, we apply visual analysis and \textit{optical character recognition} (OCR) to extract key visual features from the page screenshots (particularly the regions of the login form). The intuition is that no matter how attackers obfuscate the HTML content, the visual presentation of the page will still need to look legitimate to deceive users. Our classifier is highly accurate, with a false positive rate of 0.03 and a false negative rate of 0.06.

\textbf{Squatting Phishing Pages and Evasion.} By applying the classifier to the 657,663 squatting domains, we identified and confirmed 1,175 squatting phishing domains (857 web pages, 908 mobile pages). Our results suggest that squatting phishing pages exist but are not highly prevalent among squatting domains (0.2%). In addition, squatting phishing pages take advantage of all five domain squatting techniques to deceive users, and are used for various
targeted scams. Examples range from setting up fake Google search engines in Ukraine to scamming Uber’s truck drivers or impersonating a payroll system to scam employees. Furthermore, squatting phishing pages are more likely to adopt evasion techniques and are highly effective in evading detections. More than 90% of phishing domains successfully evaded popular blacklists such as VirusTotal (70+ blacklists), PhishTank, and eCrimeX for at least a month. Our results provide key insights into how to develop effective countermeasures.

Our chapter has three main contributions:

- First, we propose a novel end-to-end measurement framework SquatPhi to search and detect squatting phishing pages from a large number of squatting domains.\(^1\)

- Second, we perform the first in-depth analysis on squatting phishing domains in the wild. Our results provide insights into how squatting phishing pages impersonate popular brands at both the domain and content level.

- Third, we empirically characterize the evasion techniques used by squatting phishing pages. The results indicate that existing detection methods are likely to be ineffective and need to be improved.

6.3 Research Questions

Our goal is to search and identify squatting phishing pages in the wild. Through empirical measurements, we seek to understand how attackers perform impersonation to deceive users and how they perform evasion to avoid being detected. To achieve these goals, we face two major technical challenges.

First, a lack of comprehensive sources of squatting domains. It is challenging to capture a comprehensive list of squatting domains that are potentially impersonating legitimate brands and online services. More specifically, we are not looking for a specific type of domain squatting, but aim to cover all different types of squatting domains. Current phishing blacklists rarely include squatting phishing pages. Later in §6.5.1, we show that most of the reported phishing URLs in PhishTank \([33]\) do not have squatting domains.

Second, a lack of effective phishing detection tools. Phishing pages are constantly evolving. URL blacklisting is ineffective to detect zero-day phishing pages. In addition, our preliminary analysis shows that phishing pages have adopted evasion techniques that are likely to render existing detection methods ineffective (§6.5.2). An efficient and yet evasion-resilient method is needed to detect squatting phishing pages.

Our Approaches. Instead of relying on phishing blacklists, we decide to search for previous-unknown squatting phishing pages in the wild. To do so, we develop a set of

\(^1\)We open-sourced our tool at https://github.com/SquatPhish.
new tools for *squatting domain detection* and *phishing page classification*. More specifically, we select a large number of popular brands which are often targeted (impersonated) by phishing pages. Then we directly search for squatting domains that are likely to impersonate these brands from hundreds of millions of DNS records. We build a tool to effectively identify known types of squatting domains including homograph squatting, typo squatting, bit squatting, combo squatting and wrongTLD squatting.

To effectively detect phishing domains from the squatting domains, we build a novel machine learning classifier that takes advantage of image analysis and optical character recognition (OCR) to overcome page obfuscation. The classifier design is driven by empirical measurements of evasion methods used in practice (based on 4000+ manually labelled phishing pages). Once the classifier is trained, we use it to search for squatting phishing pages within a large number squatting domains. In the following, we describe each of the measurement steps and our key findings.

### 6.4 Measurement Methodology

In this section, we introduce our measurement methodology to search for candidate squatting domains. Then we introduce our data collection process to obtain their webpages (both web and mobile pages).

#### 6.4.1 Squatting Detection

At the high-level, we first select a large set of popular brands and online services which are the potential impersonation targets of squatting phishing pages. Then we detect squatting domains for each brand from massive DNS records.

**Brand Selection.** Intuitively, popular brands are attractive targets. We select domains that are ranked high by Alexa [5]. More specifically, Alexa provides 17 categories such as “business”, “games”, “health”, “finance”. For each category, we select the top 50 websites (850 domains in total). Then we search for brands that are popular targets of real-world phishing attacks. Based on the statistics from PhishTank [33], we obtain 204 brands (domains). For all the domains, we then merge some of them that share the same domain names (e.g., merging niams.nih.gov and nichd.nih.gov into nih.gov). We merge those that are co-listed by PhishTank and Alexa. In total, we have 702 unique brands (domain names) that cover a wide range of different online services.

**DNS Dataset.** Next, we search the squatting domains of the target brands within a large collection of DNS records. We obtained a snapshot of 224,810,532 DNS records from the ActiveDNS project [177] on September 6, 2017. ActiveDNS project uses multiple seeds to run active DNS probing, covering a number of top-level domains (e.g., COM, NET, ORG)
6.4. Measurement Methodology

and other lists of domain collections (e.g., Alexa Top 1M, Public Blacklists). Each record is
categorized by a domain and an IP address. We use the 224 million domain names as the
base to search for squatting domains in the next step.

Squatting Domain Identification. The most challenging step is to generate squatting
domains for the target brands. Unfortunately, the state-of-the-art tools such as DNST-
wist [17] and URLcrazy [46] are not very applicable to our case. First, existing tools are
primarily designed to generate typo squatting and bits squatting domains. They cannot
effectively handle combo squatting domains or domains that change the TLD. For example,
URL-crazy can generate facebookj.com based on typo squatting for facebook.com, but
would miss a domain facebookj.es that exists in our DNS records.

In addition, existing tools are very incomplete in detecting homograph domains. The most
important type of homograph domains is the internationalized domain names that exploit
the unicode confusion [45]. We find that tools like DNSTwist fail to map the complete list of
similar unicode characters. For example, there are 23 different unicode characters that look
similar to the letter “a” [45], but DNSTwist only catches 13 of them. These limitations will
seriously hurt our chance of capturing squatting phishing pages.

<table>
<thead>
<tr>
<th>Domain</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>faceb00k.pw</td>
<td>homograph</td>
</tr>
<tr>
<td>facebook.com</td>
<td>homograph</td>
</tr>
<tr>
<td>facebnok.tk</td>
<td>bits</td>
</tr>
<tr>
<td>facebo0ok.com</td>
<td>typo</td>
</tr>
<tr>
<td>fcaebook.org</td>
<td>typo</td>
</tr>
<tr>
<td>facebook-story.de</td>
<td>combo</td>
</tr>
<tr>
<td>facebook.audi</td>
<td>wrongTLD</td>
</tr>
</tbody>
</table>

Table 6.1: Examples of different types of squatting domains for the facebook brand.

Figure 6.2: # of squatting domain of different squatting types.

Figure 6.3: Accumulated % of squatting domains from top brands. Brands sorted by #
of domains.
Table 6.2: Top 5 brands with the most squatting domains.

<table>
<thead>
<tr>
<th>Brand</th>
<th>Squatting Domain</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>vice.com</td>
<td>39,343</td>
<td>5.98%</td>
</tr>
<tr>
<td>porn.com</td>
<td>18,149</td>
<td>2.76%</td>
</tr>
<tr>
<td>bt.com</td>
<td>16,159</td>
<td>2.46%</td>
</tr>
<tr>
<td>apple.com</td>
<td>13,465</td>
<td>2.05%</td>
</tr>
<tr>
<td>ford.com</td>
<td>12,163</td>
<td>1.85%</td>
</tr>
</tbody>
</table>

To these ends, we develop our own system to capture squatting domains given a target brand. Our system is extended from DNSTwist and URL-crazy with (1) a more complete detection of homograph domains, (2) a new module to detect wrongTLD domains, and (3) a module to handle combo squatting domains [172]. Below, we provide details on the 5 types of squatting domains our system can capture. We use the real-world examples shown in Table 6.1 to explain each squatting type. We define the 5 types to be orthogonal from each other for the ease of measurement later.

- **Homograph:** Homograph based squatting refers to squatting domains that look similar to the target domains in the visual perception. For example, two characters “rn” can be used to impersonate the character “m”. faceb00k is a homograph squatting to facebook since “00” looks similar to “oo”. More advanced homograph squatting exploit internationalized domain names (IDN). IDN utilizes Punycode encoding to convert unicode characters to ASCII. For example, xn--fcebook-8va.com is the homograph IDN. After IDN translation, the domain is presented as fàcebook.com in the browser’s address bar.

- **Typo:** Typo squatting aims to mimic the incorrectly typed domain names by users. There are several methods to generate typo squatting based on a given target domain, including insertion (adding a character), omission (deleting a character), repetition (duplicating a character) and vowel swap (re-ordering two consecutive characters). Insertion refers to add an additional character to the original domain. Omission refers to deleting a character in the domain. Repetition refers to repeating a character in the domain. Vowel swap refers to reordering two consecutive characters in the domain. For example, facebo0ok.com is a typo squatting domain by inserting “0”. fcaeb0ok.org is also a typo squatting domain by reordering “a” and “c” in the domain name.

- **Bits:** Bits squatting is to flip a bit of the domain name. A bits squatting domain is only one-bit different from the target domain. For example, facebnok.tk is bits squatting domain where one bit “o” is changed to “n”.

- **Combo:** Combo squatting is to concatenate the target domain name with other characters. The concatenation could be either attached to the head or tail. In our analysis, we particularly focus on the combo squatting with hyphens which are allowed
6.4. Measurement Methodology

<table>
<thead>
<tr>
<th>Type</th>
<th>Live Domains</th>
<th>Domains w/ Redirections</th>
<th>Redirection Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Original</td>
</tr>
<tr>
<td>Web</td>
<td>362,545</td>
<td>316,620 (87.3%)</td>
<td>6,115 (1.7%)</td>
</tr>
<tr>
<td>Mobile</td>
<td>354,297</td>
<td>308,566 (87.1%)</td>
<td>6,486 (1.8%)</td>
</tr>
</tbody>
</table>

Table 6.3: Crawling statistics. We measure the redirections to the original website and those to domain marketplaces.

<table>
<thead>
<tr>
<th>Brand</th>
<th>Domains w/ Redirection</th>
<th>Redirection Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Original</td>
</tr>
<tr>
<td>Shutterfly</td>
<td>32 (29%)</td>
<td>76 (68%)</td>
</tr>
<tr>
<td>Alliancebank</td>
<td>12 (35%)</td>
<td>21 (62%)</td>
</tr>
<tr>
<td>Rabobank</td>
<td>27 (33%)</td>
<td>48 (61%)</td>
</tr>
<tr>
<td>Priceline</td>
<td>135 (45%)</td>
<td>157 (53%)</td>
</tr>
<tr>
<td>Carfax</td>
<td>226 (50%)</td>
<td>202 (45%)</td>
</tr>
</tbody>
</table>

Table 6.4: Top brands with the highest ratio of redirections to their original websites.

in the domain name. For example, facebook-story is the combo squatting where new characters are attached to the tail of facebook with a hyphen.

- **WrongTLD:** All the above squatting techniques focus on the domain name but ignore the TLD. WrongTLD refers to domains that change the TLD but keep the domain name as the same. For example, facebook.audi belongs to the wrongTLD category since the original TLD "com" is changed to "audi".

**Domain Squatting Detection Results.** For a given brand, we search through the DNS records to look for squatting domains. For each DNS domain, we check all 5 squatting rules against the target domain. If a match is found, we label the DNS domain with the squatting type. During the domain matching, we ignore sub-domains. For example, mail.google-app.de is regarded as a combo squating domain because the domain name google-app is a combo squatting of the target brand google.

In total, we detected 657,663 squatting domains for the 702 target brands. Figure 6.2 presents the distribution of different squatting types. Clearly, combo squatting is the most common type (56%). Intuitively, combo-squatting is easy to register since one can add arbitrary words to the original domain and connect them with a hyphen. Other squatting domains such as typo-squatting would be more competitive since there are only limited ways to impersonate the target domain name.

Figure 6.3 shows that the number of squatting domains per brand is highly skewed. More specifically, we sort the brands based on their number of squatting domains, and calculate the accumulated ratio of squatting domains that the top brands generated. We observe that the top 20 brands are responsible for more than 30% of the squatting domains. Note that the
Chapter 6. Elite Phishing Domains in the Wild

<table>
<thead>
<tr>
<th>Brand</th>
<th>Domains w/ Redirection</th>
<th>Redirection Destination</th>
<th>Original</th>
<th>Market</th>
<th>Others</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zocdoc</td>
<td>29 (19%)</td>
<td>3 (2%)</td>
<td>118 (78%)</td>
<td>1 (1%)</td>
<td></td>
</tr>
<tr>
<td>Comerica</td>
<td>58 (41%)</td>
<td>0 (0%)</td>
<td>80 (57%)</td>
<td>3 (2%)</td>
<td></td>
</tr>
<tr>
<td>Verizon</td>
<td>76 (45%)</td>
<td>0 (0%)</td>
<td>83 (49%)</td>
<td>10 (6%)</td>
<td></td>
</tr>
<tr>
<td>Amazon</td>
<td>1855 (36%)</td>
<td>1 (0%)</td>
<td>2,168 (42%)</td>
<td>1,185 (23%)</td>
<td></td>
</tr>
<tr>
<td>Paypal</td>
<td>706 (56%)</td>
<td>33 (3%)</td>
<td>482 (38%)</td>
<td>35 (3%)</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.5: Top brands with highest ratio of redirections to domain marketplaces.

top brands here are not necessarily the most popular websites. Figure 6.2 presents the top 5 brands that matched the largest number of squatting domains. Typically, these domains either contains generic English word (e.g., apple, vice) or the length is too short (e.g., bt).

6.4.2 Web Crawling

To detect squatting phishing pages from a large number of squatting domains, we need to collect the web pages from each of the domains. At the high level, we aim to collect both their web version and mobile version of the pages to compare the potential differences. In addition, to assist our later classification tasks, we collect both the HTML source code and the screenshot for each page.

Crawler Design. To obtain the complete HTML content, we cannot simply query the static page using scripts like curl. Instead, we use headless browsers to load the dynamic content before saving the page. More specifically, we use the recently released Puppeteer [34], which is the headless Chrome. We have tested other alternative browsers such as Selenium [35, 119]. However, we find that Selenium is error-prone when crawling webpages at the million-level [36]. Given the high overhead of the large-scale dynamic crawling, we cannot exhaustively test all the possible browser versions and browser types. We choose a Chrome browser for its reliability. A potential limitation is that we might miss the cloaking websites that are specifically targeting IE explorer or other particular browsers. With Puppeteer, we build a distributed crawler to scan the 657K squatting domains and obtain the HTML content and take screenshots for the pages. Note that our crawling introduces almost no overhead to the target websites. Each website only receives 1-2 requests for each scan.

Web and Mobile Pages. For each domain, we capture both the web and mobile pages. We set “User-Agent” for iPhone 6 and Chrome 65 to obtain the mobile and web pages respectively. The data will help to analyze potential cloaking behavior or phishing pages that specifically target mobile or web users.
6.4. Measurement Methodology

Redirects. Our crawler follows all the redirections when visiting each domain, and records the destination URLs. We save the HTML content and the screenshots for the webpages of the destination URLs.

Distributed Crawling. To speed up our crawling efficiency, we dispatch the crawling jobs to multiple CPU cores. The original Puppeteer does not support distributed crawling. To this end, we implement our own distributed crawling by allocating a kernel-level shared memory segment count. Each time, we fork a list of children processes and utilizes `shmget` in IPC (inter process communication) to balance the workload of each process. This allows us to the maximize the usage of CPUs for the web crawling. We run the crawler on 5 machines (24 cores, 196GB RAM) and open 20 Puppeteer simultaneously.

Web Crawling Statistics. From April 01 to April 08 in 2018, we collected one snapshot of the full 657,663 domains covering both the web and mobile pages. We use this snapshot to detect squatting phishing pages. From April 09 to April 29 in 2018, we collect three additional snapshots only for the detected squatting phishing pages (one week apart between consecutive snapshots). Table 6.3 provides the statistics for the full snapshot. For the web version, we find that 362,545 domains are live and reachable. For the mobile version, we obtain data from 354,297 live domains. Overall, about 55% of the squatting domains are live during the time of crawling. Among the live domains, we find that most of them (87%) have no redirection and 13% of the domains redirect the crawler to other domains.

Interestingly, 6,115 domains (1.7%) redirect the crawler to the original target domain. This indicates that the target brands indeed purchased squatting domains to redirect their users back to the correct websites. Table 6.4 shows the top brands whose squatting domains that the highest chance to redirect users back to the original websites. Some of the top brands are related to sensitive services such as health (ZocDoc) and banking (Comerica, Alliancebank) These brands are likely to protect their users (and their reputation) by registering the squatting domains themselves.

In addition, we find some squatting domains will redirect users to some domain marketplaces where domain names are put out for sale (e.g., marketmonitor). To measure the level of such redirection, we manually compiled a list of 22 known domain marketplaces. We find that 10,734 squatting domains (3%) would redirect users to one of the domain marketplaces. Table 6.5 shows top brands whose squatting domains have the highest chance to redirect users to domain markets. Not surprisingly, a large number of squatting domains targeting popular brands such as Amazon and Paypal are listed on the market for sale. We find 2,168 Amazon squatting domains redirect to domain markets.
Chapter 6. Elite Phishing Domains in the Wild

6.5 Characterizing Evasions

So far, we have collected a large set of squatting domains and their webpages. Next we aim to systematically detect squatting phishing pages. To develop an effective phishing detection system, we need to understand whether and how phishing pages are currently and actively evading common detection methods in practice. Such knowledge will help to design more reliable features to capture squatting phishing pages. In the following, we first collect and label ground-truth phishing pages and then perform a preliminary analysis of their evasion techniques.

6.5.1 Ground Truth Phishing Pages

We first collect ground-truth phishing pages to understand evasion and train our machine learning classifiers. Here, we don’t want to use any existing automated phishing detection tools to label the ground-truth since existing tools may be vulnerable to evasions. Instead, we rely on user reported and manually verified phishing pages as ground-truth. More specifically, we choose PhishTank [33], an online service that leverages crowdsourcing to collect phishing URLs. Any Internet users can submit phishing URLs and others can help to verify if the reported pages are truly phishing.

PhishTank Crawling. From February 2 to April 10 in 2018, we set up a crawler to crawl the phishing URLs under all 204 brand names provided by PhishTank. We ignore the brand named “other” since it does not specify the target brand. For each brand, our crawler checked the latest list 5 times a day to make sure we don’t miss any newly submitted URLs. We focus on URLs that have been verified as phishing and URLs that are marked as “active”. This allows us to immediately crawl the live phishing webpages under the reported URL. Same as before, for each Phishing URLs, we use a dynamic crawler to obtain its web and mobile pages and take screenshots for both pages.

In total, we crawled 6,755 unique phishing URLs under 138 brands. The other 66 brands do not have any URL submissions during our data collection period. As shown in Figure 6.4,
6.5. Characterizing Evasions

<table>
<thead>
<tr>
<th>Brand</th>
<th># of URLs</th>
<th>Percent (%)</th>
<th>Valid Phishing</th>
</tr>
</thead>
<tbody>
<tr>
<td>PayPal</td>
<td>1306</td>
<td>19.3</td>
<td>348</td>
</tr>
<tr>
<td>Facebook</td>
<td>1059</td>
<td>15.6</td>
<td>734</td>
</tr>
<tr>
<td>Microsoft</td>
<td>580</td>
<td>8.6</td>
<td>285</td>
</tr>
<tr>
<td>Santander UK</td>
<td>336</td>
<td>5.0</td>
<td>30</td>
</tr>
<tr>
<td>Google</td>
<td>218</td>
<td>3.2</td>
<td>95</td>
</tr>
<tr>
<td>Ebay</td>
<td>189</td>
<td>2.8</td>
<td>90</td>
</tr>
<tr>
<td>Aode</td>
<td>166</td>
<td>2.4</td>
<td>79</td>
</tr>
<tr>
<td>Dropbox</td>
<td>150</td>
<td>2.2</td>
<td>70</td>
</tr>
<tr>
<td>SubTotal</td>
<td>4004</td>
<td>59.1</td>
<td>1731</td>
</tr>
</tbody>
</table>

Table 6.6: Top 8 brands in PhishTank cover 4004 phishing URLs (59.1%). Manual verification shows that 1731 pages are true phishing pages.

the number of phishing pages per brand is highly skewed. The top 8 popular brands cover 4004 phishing URLs which counts for 59% of total phishing URLs.

**Popularity and Squatting.** To provide contexts for the phishing URLs, we first examine the ranking of their domains on Alexa top 1 Million list. As shown in Figure 6.5, the vast majority (4749, 70%) of the phishing URLs are ranked beyond the Alexa top 1 million. This suggests most phishing pages are hosted on unpopular domains. A further analysis shows that `000webhostapp` is most frequently used hosting domains for phishing pages (914 URLs) followed by `sites.google` and `drive.google` (140 URLs). The result suggests web hosting services have been abused by phishing.

We then analyze the squatting domains in the phishing URLs. As shown in Figure 6.6, the majority of phishing URLs are not squatting phishing — 6,156 (91%) of phishing URLs did not use squatting domains. In addition to the combo-squatting domains, we find one homograph squatting `google.online` for `google`, one typo squatting `paypals.center` for `paypal`. There is no bits squatting or wrongTLD squatting in the PhishTank. This confirms that we cannot rely on phishing blacklists to study squatting phishing.

**Ground Truth Labeling.** Although the phishing URLs from PhishTank have been “validated”, it is possible some of phishing pages have been replaced or taken-down when we crawl the pages. To this end, we cannot simply label all the crawled pages as “phishing”. To obtain the ground-truth label, we select the top 8 brands (4,004 URLs, 59.1%) to manually examine the crawled pages (screenshots). As shown in Table 6.6, surprisingly, it turns out a large number of pages are no longer considered as phishing pages during the time of crawling. Only 1,731 out of 4,004 (43.2%) are still phishing pages. The rest 2,273 pages are no longer phishing pages (benign). Recall that our crawler has been monitoring the newly submitted URLs to PhishTank and immediately crawled their pages. The results suggest that phishing pages have a very short lifetime. Many phishing URLs have been taken-down or replaced with legitimate pages before the URLs are listed on PhishTank.
Chapter 6. Elite Phishing Domains in the Wild

Figure 6.7: An example of page layout obfuscation of phishing pages (paypal).

Figure 6.8: Average Image hash distance and standard variance for phishing pages of different brands.

6.5.2 Evasion Measurement

Based on the ground-truth data, we next examine the common evasive behavior of phishing pages. We will use the measurement results to derive new features to more robust phishing page detection. Our evasion measurement focuses on three main aspects: the image layout, the string text in the source code, and obfuscation indicators in the javascript code. These are common places where adversaries can manipulate the content to hide its malicious features, while still giving the web page a legitimate look. For this analysis, we focus on the web version of the pages. We find that 96% of the pages on PhishTank have the same underlying HTML sources for both the web and mobile versions. This indicates that the most attackers did not show different pages to the web and mobile users (i.e. no cloaking).

Layout Obfuscation. Many phishing detection methods assume that the phishing pages will mimic the legitimate pages of the target brands. As a result, their page layout should share a high-level of similarity [213]. Phishing detection tools may apply some fuzzy hashing functions to the page screenshots and match them against the hash of the real pages. To examine the potential evasions against page layout matching, we compute the Image hash [26] to compare the visual similarity of the phishing pages and the real pages of the target brands. The (dis)similarity is measured by the hamming distance between two
image hashes.

We find that layout obfuscation is widely applied, and phishing pages often change their layout greatly to evade detection. Figure 6.7 shows a real example in our dataset for brand **paypal**. The left-most page is the official **paypal** page. The other 3 pages are phishing pages with different image hash distances 7, 24 and 36 respectively compared to the real pages. With a distance of 7, the phishing page is still visually similar to the original page. When the distance goes to 24 and 36, the pages look different from the original pages but still have a legitimate looking. Those pages would be easily missed by visual similarity based detectors.

Figure 6.8 shows the average image hash distance to the original pages for all phishing pages of different brands. We show that most brands have an average distance around 20 or higher, suggesting that layout obfuscation is very common. In addition, different brands have a different level of visual similarity, which makes it difficult to set a universal threshold that works for all the brands. These evasion steps would likely to render visual similarity based detection methods ineffective.

**String Obfuscation.** String obfuscation is hiding important text and keywords in the HTML source code. For example, attackers may want to hide keywords related to the target brand names to avoid text-matching based detection [169]. For example, in a phishing page that impersonates **paypal**, we find that the brand name string is obfuscated as “PayPaI”, where the “l” (the lower case of “L”) is changed to “I” (the upper case of “i”). Another common technique is to delete all related text about the brand name **paypal** but instead put the text into images to display them to users. From the users’ perspective, the resulting page will still look similar.

We perform a simple measurement of text string obfuscation by looking for the brand name in the phishing pages’ HTML source. Given a phishing page (and its target brand), we first extract all the texts from the HTML source. If the target brand name is not within the texts, then we regard the phishing page as a string obfuscated page. Table 6.7 shows the percentage of string obfuscated pages for each brand. For example, 70.2% of **microsoft** phishing pages are string obfuscated. 35.3% of **facebook** phishing pages are string obfuscated. This suggests that simple string matching is less likely to be effective.

**Code Obfuscation.** Javascript code may also apply obfuscation to hide their real purposes. This is a well-studied area and we use known obfuscation indicators to measure the level of code obfuscation in the phishing pages. Obfuscation indicators are borrowed from FrameHanger [278]. According to previous studies [165, 308], string functions (e.g., `fromChar` and `charCodeAt`), dynamic evaluation (e.g., `eval`) and special characters are heavily used for code obfuscation. For each phishing page, we download and parse the JavaScript code into an AST (abstract syntax tree). We then use AST to extract obfuscation indicators.

Table 6.7 presents the percentage of phishing pages that contain obfuscation indicators.


<table>
<thead>
<tr>
<th>Brand</th>
<th>String Obfuscated</th>
<th>Code Obfuscated</th>
</tr>
</thead>
<tbody>
<tr>
<td>Santander</td>
<td>30 (100%)</td>
<td>4 (13.3%)</td>
</tr>
<tr>
<td>Microsoft</td>
<td>200 (70.2%)</td>
<td>127 (44.6%)</td>
</tr>
<tr>
<td>Adobe</td>
<td>38 (48.1%)</td>
<td>15 (18.9%)</td>
</tr>
<tr>
<td>Facebook</td>
<td>259 (35.3%)</td>
<td>342 (46.6%)</td>
</tr>
<tr>
<td>Dropbox</td>
<td>16 (22.9%)</td>
<td>1 (1.5%)</td>
</tr>
<tr>
<td>PayPal</td>
<td>61 (17.5%)</td>
<td>140 (40.2%)</td>
</tr>
<tr>
<td>Google</td>
<td>10 (10.5%)</td>
<td>11 (11.6%)</td>
</tr>
<tr>
<td>Ebay</td>
<td>8 (8.9%)</td>
<td>9 (10.0%)</td>
</tr>
</tbody>
</table>

Table 6.7: String and code obfuscation in phishing pages.

Since we focus on strong and well-known indicators only, the results are likely to represent a lower bound of code obfuscation in phishing. For example, we find that some Adobe phishing pages adopt php script “action.php” for login forms. The script is invoked from a php file stored in a relative path. Automated analysis of php code (in a relative path) to detect obfuscation is a challenging problem itself.

### 6.6 Machine-Learning Detection

After understanding the common evasion techniques, we now design a new machine learning based classifier to detect squatting phishing pages. The key is to introduce more reliable features. Below, we first introduce our feature engineering process and then we train the classifier using the ground-truth data obtained from PhishTank. Finally, we present the accuracy evaluation results.

#### 6.6.1 Feature Engineering

Based on the analysis in §6.5.2, we show that visual features, text-based features and javascript based features can be evaded by obfuscations. We need to design new features to compensate for existing ones. More specifically, we are examining squatting domains that are already suspicious candidates that attempt to impersonate the target brands. Among these suspicious pages, there are two main hints for phishing. First, the page contains some keywords related to the target brands either in the form of plaintext, images, or dynamically generated content by Javascripts. Second, the page contains some “forms” to trick users to enter important information. For example, this can be a login form to collect passwords or payment forms to collect credit card information.

To overcome the obfuscations, our intuition is that no matter how the attackers hide the keywords in the HTML level, the information will be visually displayed for users to complete
the deception. To this end, we extract our main features from the screenshots of the suspicious pages. We use optical character recognition (OCR) techniques to extract text from the page screenshots to overcome the text and code level obfuscations. In addition, we will still extract traditional features from HTML considering that some phishing pages may not perform evasion. Finally, we consider features extracted from various submission “forms” on the page. All these features are independent from any specific brands or their original pages. This allows the classifier to focus on the nature of phishing.

Image-based OCR Features. From the screenshots, we expect the phishing page to contain related information in order to deceive users. To extract text information from a given page screenshot, we use OCR (Optical character recognition), a technique to extract text from images. With the recent advancement in computer vision and deep learning, OCR’s performance has been significantly improved in the recent years. We use the state-of-the-art OCR engine Tesseract [43] developed by Google. Tesseract adopts an adaptive layout segmentation method, and can recognize texts of different sizes and on different backgrounds. According to Google, the recent model has an error rate below 3% [44], which we believe this is acceptable for our purpose. By applying Tesseract to the crawled screenshots, we show that Tesseract can extract text such as “paypal” and “facebook” directly from the logos areas of the screenshots. More importantly, from the login form areas, it can extract texts such as “email” and “password” from the input box, and even “submit” from the login buttons. We treat the extracted keywords as OCR features.

Text-based Lexical Features. We still use text based features from HTML to complement OCR features. To extract the lexical features, we extract and parse the text elements from the HTML code. More specifically, we focus on the following HTML tags: h tag for all the texts in the headers, p tag for all the plaintexts, a tag for texts in the hyperlinks, and title tag for the texts in the title of the page. We do not consider texts that are dynamically generated by JavaScript code due to the high overhead (which requires dynamically executing the javascript in a controlled environment). We treat these keywords as lexical features.

Form-based Features. To extract features from data submission forms, we identify forms from HTML and collect their attributes. We focus on 4 form attributes: type, name, submit and placeholder. The placeholder attribute specifies a short hint for the input box. Often cases, placeholder shows hints for the “username” and “password” in the phishing pages, e.g., “please enter your password”, “phone, email or username”. The name attribute specifies the name of the button. We treat the texts extracted from the form attributes as features. We also consider the number of forms in the HTML document as a feature.

Features that We Did Not Use. Prior works have proposed other features but most of which are not applicable for our purpose. For example, researchers of [61, 113, 303] also considered OCR and lexical features, but the underlying assumption is that phishing
sites share a high level similarity with the real sites (visually or keyword-wise). However, this assumption is not necessarily true given the evasion techniques and the large variances of phishing pages (§6.5.2). In addition, Cantina [322] and Cantina+ [303] propose to query search engines (e.g., Google) using the keywords of the suspicious web pages to match against the real sites. However, these features are too expensive to obtain given the large scale of our dataset. To these ends, the features we chose in this chapter (e.g., keywords from logos, login forms, and other input fields) are lightweight and capture the essentials of a phishing page which are difficult to tweak without changing its impression to a user.

**Discussions on the Feature Robustness.** So far, we haven’t seen any real-world phishing pages that attempt to evade the OCR engine. Future attackers may attempt to add adversarial noises to images to manipulate the OCR output. However, technically speaking, evading OCR features are difficult in the phishing contexts. First, unlike standard image classifiers that can be easily evaded [91, 139, 145, 198, 214, 234, 309], OCR involves a more complex segmentation and transformation process on the input images before the text extraction. These steps make it extremely difficult to reverse-engineer a blackbox OCR engine to perform adversarial attacks. A recent work confirms that it is difficult to evade OCR in a blackbox setting [263]. Second, specifically for phishing, it is impossible for attackers to add arbitrary adversarial noises to the whole screenshots. Instead, the only part that attackers can manipulate is the actual images loaded by the HTML. This means texts of the login forms and buttons can still be extracted by OCR or from the form attributes. Finally, for phishing, the key is to avoid alerting users, and thus the adversarial noise needs to be extremely small. This further increases the difficulty of evasion. Overall, we believe the combination of OCR features and other features helps to increases the performance (and the robustness) of the classifiers.

### 6.6.2 Feature Embedding and Training

After the raw features are extracted, we need to process and normalize the features before used them for training. Here, we apply NLP (natural language processing) to extract meaningful keywords and transform them into training vectors.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>False Positive</th>
<th>False Negative</th>
<th>AUC</th>
<th>ACC</th>
</tr>
</thead>
<tbody>
<tr>
<td>NaiveBayes</td>
<td>0.50</td>
<td>0.05</td>
<td>0.64</td>
<td>0.44</td>
</tr>
<tr>
<td>KNN</td>
<td>0.04</td>
<td>0.10</td>
<td>0.92</td>
<td>0.86</td>
</tr>
<tr>
<td>RandomForest</td>
<td>0.03</td>
<td>0.06</td>
<td>0.97</td>
<td>0.90</td>
</tr>
</tbody>
</table>

Table 6.8: Classifiers’ performance on ground-truth data.

Tokenization and Spelling Checking. We first use NLTK [76], a popular NLP toolkit to tokenize the extracted raw text and then remove the stopwords [42]. Since the OCR engine itself would make mistakes, we then apply spell checking to correct certain typos from OCR. For example, Tesseract sometimes introduces errors such as “passwod”, which can be easily corrected to “password” by a spell checker. In this way, we obtain a list of keywords for each page.

Feature Embedding. Next, we construct the feature vector. For numeric features (e.g., number of forms in HTML), we directly append them to the feature vector. For keyword-related features, we use the frequency of each keyword in the given page as the feature value. During training, we consider keywords that frequently appear in the ground-truth phishing pages as well as the keywords related to all the 766 brand names. The dimension of the feature vector is 987 and each feature vector is quite sparse.

Classifiers. We tested 3 different machine learning models including Naive Bayes, KNN and Random forest. These models are chosen primarily for efficiency considerations since the classifier needs to quickly process millions of webpages.

6.6.3 Ground-Truth Evaluation

We use the ground-truth phishing pages from PhishTank to evaluate the classifier’s performance. The classifier is trained to detect whether a page is phishing (positive) or not (negative). Recall that in § 6.5.1, there is no major difference in the HTML code for web and mobile pages, we only use the web version to perform the training.

The ground-truth dataset contains 1731 manually verified phishing pages from PhishTank. The benign categories contain 3838 webpages from two sources: the first part of 2273 benign pages were manually identified from the PhishTank dataset (§6.5.1); The second part of benign pages come from the webpages of the 1.6 million squatting domains (§6.4.2). We randomly sampled and manually verified 1565 benign pages. Due to the time-consuming nature of manual annotation, we only introduce the most “easy-to-confuse” benign pages (i.e., those under squatting domains and those incorrectly reported as phishing). We did not include the “obviously benign pages” so that the classifiers can be more focused to distinguish the benign pages from the squatting domain set.

Table 6.8 shows the results of 10-fold cross-validation. We present the false positive rate,
false negative rate, area under curve (AUC) and accuracy (ACC). We show that Random Forest has the highest AUC (0.97), with a false positive rate of 0.03 and a false negative rate 0.06. The classifier is highly accurate on the ground-truth dataset. Figure 6.9 presents the ROC curve of three algorithms. Random Forest achieves the best performance, and will be used to detect squatting phishing domains from the squatting domains.

6.7 Squatting Phishing in the Wild

In this section, we apply our classifier to detect squatting phishing pages in the wild. We first describe our detection results and manually the confirm the flagged phishing pages. Then we analyze the squatting phishing pages to answer the following questions. First, how prevalent are phishing pages among the squatting domains? Second, what are the common attacks that squatting phishing pages are used for, and what types of squatting techniques are used? Third, are squatting phishing pages more evasive? How quickly can squatting phishing pages be detected or blacklisted?

6.7.1 Detecting Squatting Phishing Pages

We apply the Random Forest classifier to the collected web and mobile pages from the squatting domains. As shown in Table 6.9, the classifier detected 1,224 phishing pages for the web version, and 1,269 phishing pages for the mobile version. Comparing to the 657,663 squatting domains, the number of squatting phishing pages are relatively small (0.2%).
6.7. Squatting Phishing in the Wild

Table 6.10: 15 example brands and verified phishing pages.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Google</td>
<td>6,801</td>
<td>112</td>
<td>97</td>
<td>105 (94%)</td>
<td>89 (92%)</td>
</tr>
<tr>
<td>Facebook</td>
<td>3,837</td>
<td>21</td>
<td>24</td>
<td>18 (86%)</td>
<td>19 (80%)</td>
</tr>
<tr>
<td>Apple</td>
<td>13,465</td>
<td>20</td>
<td>22</td>
<td>8 (40%)</td>
<td>16 (72%)</td>
</tr>
<tr>
<td>BitCoin</td>
<td>1,378</td>
<td>19</td>
<td>17</td>
<td>16 (84%)</td>
<td>16 (94%)</td>
</tr>
<tr>
<td>Uber</td>
<td>5,963</td>
<td>16</td>
<td>16</td>
<td>11 (69%)</td>
<td>11 (69%)</td>
</tr>
<tr>
<td>Youtube</td>
<td>3,162</td>
<td>16</td>
<td>15</td>
<td>4 (25%)</td>
<td>12 (80%)</td>
</tr>
<tr>
<td>PayPal</td>
<td>2,330</td>
<td>14</td>
<td>17</td>
<td>7 (50%)</td>
<td>7 (41%)</td>
</tr>
<tr>
<td>Citi</td>
<td>5,123</td>
<td>10</td>
<td>19</td>
<td>8 (80%)</td>
<td>11 (58%)</td>
</tr>
<tr>
<td>Ebay</td>
<td>3,109</td>
<td>8</td>
<td>8</td>
<td>5 (63%)</td>
<td>5 (63%)</td>
</tr>
<tr>
<td>Microsoft</td>
<td>3,039</td>
<td>7</td>
<td>2</td>
<td>5 (71%)</td>
<td>2 (100%)</td>
</tr>
<tr>
<td>Twitter</td>
<td>1,378</td>
<td>7</td>
<td>5</td>
<td>4 (57%)</td>
<td>5 (100%)</td>
</tr>
<tr>
<td>DropBox</td>
<td>516</td>
<td>5</td>
<td>3</td>
<td>3 (60%)</td>
<td>2 (67%)</td>
</tr>
<tr>
<td>GitHub</td>
<td>503</td>
<td>6</td>
<td>4</td>
<td>5 (83%)</td>
<td>2 (50%)</td>
</tr>
<tr>
<td>ADP</td>
<td>3,305</td>
<td>6</td>
<td>7</td>
<td>3 (50%)</td>
<td>3 (43%)</td>
</tr>
<tr>
<td>Santander</td>
<td>567</td>
<td>1</td>
<td>1</td>
<td>1 (100%)</td>
<td>1 (100%)</td>
</tr>
</tbody>
</table>

**Manual Verification.** After the classification, we manually examined each of the detected phishing pages to further remove classification errors. During our manual examination, we follow a simple rule: if the page impersonates the trademarks of the target brands and if there is a form to trick users to input personal information, we regard the page as a phishing page. As shown in Table 6.9, after manual examination, we confirmed 1,175 domains are indeed phishing domains. Under these domains, there are 857 web phishing pages which count for 70.0% of all flagged web pages by the classifier. In addition, we confirmed even more mobile phishing pages (908) which count for 72.0% of all flagged mobile pages.

In Table 6.10, we present 15 example brands and the number of confirmed squatting phishing pages. We show the detection accuracy of the classifier is reasonably high for popular brands such as Google, Facebook, and Microsoft. However, the classifier is more likely to make mistakes on brands such as Paypal, Twitter, and Uber. Our manual analysis shows that the errors largely come from legitimate pages that contain some submission forms (e.g., survey text boxes to collect user feedback) or third-party plugins of the target brands (e.g., plugins for supporting payments via PayPal, Twitter “share” icons, Facebook “Like” buttons). The results suggest that the classifier trained on the ground-truth dataset is still not perfect. Since the testing data is orders of magnitude larger, it is possible that certain variances are not captured during the small-scale training. A potential way of improvement is to feed the newly confirmed phishing pages back to the training data to re-enforce the classifier training (future work).

**Targeted Brands.** As shown in Table 6.9, the confirmed phishing pages are targeting
Chapter 6. Elite Phishing Domains in the Wild

281 brands (247 brands on the web, and 255 brands on the mobile version). The rest of the 421 brands do not have squatting phishing pages under their squatting domains. Figure 6.10 shows the number of verified phishing pages for each brand. We show the vast majority of brands have fewer than 10 squatting phishing pages. Most brands are impersonated by tens of squatting phishing pages.

To illustrate the brands that are highly targeted by squatting phishing domains, we plot Figure 6.12. We observe that google standout as the mostly impersonated brands with 194 phishing pages across web and mobile. Google's number if much higher than the second and third brands which all have 40 or below squatting phishing pages. We observe the popular brands such as ford, facebook, bitcoin, amazon, and apple are among the heavily targeted brands. Figure 6.13 shows a few examples squatting phishing pages that mimic the target brands at both the content level and the domain level.

Mobile vs. Web. An interesting observation is that mobile and web does not have the same number of phishing pages. There are more mobile phishing pages. This indicates a cloaking behavior — the phishing websites only respond to certain types of user devices. Among the 1175 phishing domains, only 590 domains have both web and mobile phishing pages. 318 domains only show phishing pages to mobile users but not to web users; 267 domains return phishing pages to web users only. A possible reason for attackers to target mobile users is that mobile browsers do not always show the warning pages like the web browsers. During manual analysis, we used a Chrome browser on the laptop and a mobile Chrome browser to visit the confirmed phishing domains. The laptop Chrome is more likely to show the alert page compared to the mobile browser for the same domain. We also tested the laptop and mobile version of Safari and observed the same phenomenon.
As a related note, recent studies show that mobile browsers’ UI design could make users more vulnerable to phishing [203, 226]. For example, mobile browsers often cannot fully display very long URLs in the address bar, and thus only show the leftmost or the rightmost part to users. This design limits a user’s ability to examine the domain name of the (phishing) URL. In our case, we only find a few long domain names from the 1175 phishing domains. For example, the longest domain name is “buy-bitcoin-with-paypal-paysafecard-credit-card-ukash.com” which has 57 characters.

IP Location. We further examine the geolocation of the IP addresses of the phishing domains. In total, we are able to look up the geolocation of 1,021 IP addresses, hosted in 53 different countries. Figure 6.14 shows the IP distribution in different countries and we highlight the top countries with the most IP addresses. These phishing sites are widely spread all over the world. The U.S. has most of the sites, followed by Germany (DE).

Domain Name Registration. Finally, we obtain the whois records of the phishing domain names and examine their registration time and registrars. As shown in Figure 6.15, most of the squatting phishing domain names were registered within the recent 4 years. Based on the whois records, only 738 domains contain the registrar information. We find that out of 121 different registrar institutions, the most popular registrar is godaddy.com with 157 squatting phishing domain names.

### 6.7.2 Squatting Types & Case Studies

Next, we examine the squatting methods used by squatting phishing domains. As shown in Figure 6.11, there are squatting phishing pages under each every squatting method. It is not too surprising that combo squatting domains contain the largest number of phishing pages since they are less competitive to register, i.e., attackers can add arbitrary strings to the target brand names. We find over 200 phishing pages within homograph squatting domains, bits squatting domains and typo squatting domains, which are more difficult to register. Table 6.14 shows a few examples of the phishing domains of different squatting types. We
select 6 examples and present their screenshots in Figure 6.13, and infer the motivations behind the squatting phishing pages.

**Fake Search Engine.** Figure 6.13a presents an interesting example of bits squatting. The phishing domain “goofle.com.ua” is trying to impersonate Google’s search engine hosted in Ukraine “google.com.ua”, by changing one character “g”. A possible motivation of this page is to perform censorship to monitor what searching queries that Ukraine citizens are performing. Another (more likely) motivation is that this website impersonates Google search to serve specific advertisements to users. Through manual examination, we find that the fake search engine not only displays more advertisements, but the advertisements are also different from those on the real site, given the same searching query (the searching results are relatively consistent).

**Offline Scam.** Figure 6.13b shows an example of combo squatting. The squatting phishing domain is “go-uberfreight.com”, which impersonates Uber Freight, a new service of Uber to connect truck drivers with shippers. The official site is freight.uber.com. The purpose of the phishing page is likely to steal truck drivers’ Uber accounts. Note that truck drivers’ accounts are very difficult to register which takes background checks and virtual/on-site interviews. It is possible that the attacker is trying to steal truck driver’s account for offline scams, for example, to impersonate an Uber truck driver to pick up and steal valuable goods. Another related example is shown in Figure 6.13e where the phishing domain “driveforuber.com” is impersonating the official Uber site “drive.uber.com”.

**Payroll Scam.** Figure 6.13d shows a payroll scam on ADP. ADP offers payroll services for employees of various companies. ADP’s official mobile domain is “mobile.adp.com”. The phishing page “mobile-adp.com” is impersonating the mobile page of ADP through combo squatting. Interestingly, the login form on the phishing page is dynamically loaded by a JavaScript. We find that the login form will show up only if a user did not have an adblocker.

**Tech Support Scam.** Figure 6.13c shows a tech support scam where a combo-squatting domain “live-microsoftsupport.com” is trying to impersonate the online support website of Microsoft “support.microsoft.com”. The page either tries to compromise a user’s Microsoft account or trick the user to call the listed phone number. For example, scammers behind the phone may guide the victim to install malware or pay the “service fee” [216].

**Stealing Payment Accounts.** More commonly, squatting phishing pages aim to compromise user accounts at payment services. For example, Figure 6.13f is a phishing page that impersonates Citizens Bank’s official page “citizenslc.com”. The phishing domain is a combo squatting domain “securemail-citizenslc.com”.


6.7. Squatting Phishing in the Wild

<table>
<thead>
<tr>
<th>Type</th>
<th>Layout Obfuscation</th>
<th>String Obfuscation</th>
<th>Code Obfuscation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Squatting-Web</td>
<td>28.4 ± 11.8</td>
<td>68.1%</td>
<td>34.0%</td>
</tr>
<tr>
<td>Squatting-Mobile</td>
<td>28.6 ± 11.6</td>
<td>68.2%</td>
<td>35.3%</td>
</tr>
<tr>
<td>Non-Squatting</td>
<td>21.0 ± 12.3</td>
<td>35.9%</td>
<td>37.5%</td>
</tr>
</tbody>
</table>

Table 6.11: Phishing pages that adopted evasion techniques.

<table>
<thead>
<tr>
<th>Blacklist</th>
<th>PhishTank</th>
<th>VirusTotal</th>
<th>eCrimeX</th>
<th>Not Detected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domains</td>
<td>0 (0.0%)</td>
<td>100 (8.5%)</td>
<td>2 (0.2%)</td>
<td>1,075 (91.5%)</td>
</tr>
</tbody>
</table>

Table 6.12: Detected squatting phishing pages by popular blacklists. VirusTotal contains 70+ blacklists.

6.7.3 Evasion

We next examine whether and how squatting phishing pages would perform evasion against common detection techniques.

Squatting vs. Non-Squatting Phishing. We first examine whether squatting phishing pages are more likely to adopt evasion techniques. In Table 6.11, we directly compare the verified squatting phishing pages with non-squatting phishing pages labelled in PhishTank (URLs without squatting domains). As shown in Table 6.11, squatting phishing pages have a higher level of layout obfuscation. In addition, there is a higher ratio of squatting phishing pages that adopted string obfuscation (68.1%–68.2%) than that of non-squatting phishing pages (35.9%). Code obfuscation is less common among squatting phishing pages.

Evading Popular Blacklists. The phishing pages detected by our system are largely previous-unknown phishing pages. To examine how likely they can evade existing blacklist, we perform a quick test. As shown in Table 6.12, we run the list of verified squatting phishing domains against several popular phishing blacklists in May 2018. First, we checked the PhishTank and find that only 2 of our squatting phishing domains have been reported (0.1%). Then we query VirusTotal [47], which contains over 70 different blacklists. These 70 blacklists collectively marked 110 (8.2%) of squatting phishing domains. Finally, examine eCrimeX [18], a phishing blacklist maintained by the Anti Phishing Work Group (APWG). Their phishing URLs are gathered from a large number organizations around the globe. Through collaboration, we obtained 335,246 phishing URLs reported during April 2017 to April 2018. In total, eCrimeX marked 4 squatting phishing domains (0.2%). Collectively these blacklists only detected 8.4% of the squatting phishing pages, which means 91.5% of the phishing domains remain undetected for at least a month. As a comparison, a recent study [144] shows that phishing pages hosted on compromised web servers typically last for less than 10 days before they are blacklisted. This suggests that squatting phishing domains are much more difficult to detect.
Table 6.13: The liveness of phishing pages on different dates.

<table>
<thead>
<tr>
<th>Domain</th>
<th>April 01</th>
<th>April 08</th>
<th>April 22</th>
<th>April 29</th>
</tr>
</thead>
<tbody>
<tr>
<td>facecook.mobi</td>
<td>Live</td>
<td>Live</td>
<td>Live</td>
<td>Live</td>
</tr>
<tr>
<td>facebook-c.com</td>
<td>Live</td>
<td>Live</td>
<td>Live</td>
<td>Live</td>
</tr>
<tr>
<td>face-book.online</td>
<td>Live</td>
<td>Live</td>
<td>Live</td>
<td>Live</td>
</tr>
<tr>
<td>facebook-sigin.com</td>
<td>Live</td>
<td>Live</td>
<td>Live</td>
<td>Live</td>
</tr>
<tr>
<td>faceboolk.ml</td>
<td>Live</td>
<td>Live</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>tacebook.ga</td>
<td>Live</td>
<td>Live</td>
<td>-</td>
<td>Live</td>
</tr>
</tbody>
</table>

Life-time of Squatting Phishing Pages. We also measure the longevity of phishing pages. Recall that for domains that are classified as phishing in the first snapshot, we continue to crawl their webpages every week for a month. For each snapshot, we re-apply our classifier to their pages and examine if they are still classified as phishing. The results are shown in Figure 6.16. Most pages (about 80%) still remain alive after at least a month. Only a small portion of the pages has been down after 1-2 weeks. This again confirms that squatting phishing pages are difficult to detect and take-down.

Table 6.13 presents the liveness of 6 phishing pages that impersonate Facebook. An interesting domain is tacebook.ga. In the third snapshot, we find that the webpage under this domain has been replaced with a benign page (manually verified). However, in the fourth snapshot, the phishing page come back again.

6.8 Discussion

Detecting Squatting Phishing in Practice. In this chapter, we demonstrate a systematic approach to search and detect squatting phishing pages. With a deep-level impersonation, squatting phishing domains do not come with a large number, but are likely to be used for highly targeted attacks. Our results have shown that squatting phishing pages are difficult to detect and take down — 91.6% of them are still alive after at least a month.

Our system SquatPhi can be used in two ways. First, any third-party organizations can set up a scanner to constantly monitor the squatting domains for a broad range of brands to capture squatting phishing domains. Crowdsourcing efforts can be introduced to speed up the manual verification process. Note that we are searching needle in a haystack by narrowing down the target from hundreds of thousands squatting domains to several hundreds phishing candidates, which are then manageable for manual investigation. Second, individual online services can set up their own dedicated scanner to search for squatting phishing pages that impersonate their brands. For example, PayPal can keep monitoring the newly registered domain names to the DNS to identify PayPal related squatting domains and classify squatting phishing pages. The classifier can be potentially much more accurate if it is customized.
6.9. Related Work

for one specific brand. We have open-sourced our tool at https://github.com/SquatPhish to propel future search in the community.

**Reporting Phishing Websites.** In September 2018, we checked PhishTank, eCrimeX and VirusTotal again. Among the 1,175 verified squatting domains, 1,075 of them are still online, and only 60 (5.1%) of them are blacklisted. We then reported the rest 1,015 phishing websites to Google safe browsing (under VirusTotal). Like most blacklists, Google safe browsing does not support batch reporting, and has strict rate limits and CAPTCHAs to prevent abuse. We submitted the malicious URLs one by one manually.

**Our Limitations.** Our study has a number of limitations. First, our crawler only sets two profiles for a specific version of iPhone (mobile) and Chrome (web). It is possible that we might have missed phishing pages that perform cloaking, e.g., those that only target Microsoft Explorer users. Second, our measurement primarily focuses on “popular brand” based on Alexa ranking. As a future work, we can extend our measurement scope to specifically cover the web domains of government agencies, military institutions, universities, and hospitals to detect squatting phishing pages targeting important organizations. Third, technically, it is difficult to evade a blackbox OCR engine while creating highly deceptive phishing pages (see §6.6.1). Reverse-engineering OCR for adversarial attacks is out of the scope of this chapter. We leave more detailed explorations to future work. Finally, we did not directly compare our phishing classifier with existing tools such as Cantina [322] and Cantina+ [303]. This is because most existing works did not open-source their tool, and some of their features are too expensive to obtain for large-scale datasets. In this chapter, we open-sourced our tool to ensure the reproducibility of the results.

### 6.9 Related Work

**Squatting Domains Identification.** Previous works have studied different types of squatting techniques [63, 217, 271]. For example, More et al. [217] measured typo squatting by generating a list of plausible misspellings of popular domains. Nikiforakis et al. [224] measured the bit squatting by generating a single bit-flip for a valid domain. Holgers et al. [147] characterized homograph squatting through character substitutions. Kinti et al. [172] measured combo squatting by searching domain keywords from DNS records. In this chapter, we focus on aggregating and improving existing squatting methods to search for squatting phishing attacks.

**Phishing Webpage Detection.** A plethora of research has focused on blacklisting or content-based detection methods. For example, PhishTank [33] leverages crowdsourcing to collect phishing URLs that Internet users encountered. PhishEye [144] proposed to use honeypots to monitor live phishing pages. Other detection methods are based on visual similarities [213, 297] or lexical URL properties [79, 96, 205] to detect phishing pages. For
example, DeltaPhish [99] detects compromised websites by comparing the page structure similarities. Cantina and Cantina+ [303, 322] are based on the keyword frequency and page rank information. Marchal et al. [208] also use keyword frequency in the HTML pages. In this chapter, we show how today’s phishing pages, especially squatting phishing pages, have adopted evasion techniques that are likely to render existing detectors ineffective. A recent system Meerkat [81] uses deep learning models to analyze visual elements in webpages to detect compromised websites. Our approach is different since we use OCR to extract the text from the screenshots rather than focusing on the visual elements. Note that researchers of [61, 113] used OCR to extract keywords and query search engines to match again the real sites. However, this design still assumes phishing sites are similar/identical to the target sites, which is not necessarily true given the big variances introduced by the evasion techniques. Instead, we focus on more generic keywords extracted from logos, login forms, and other input fields to model the “phishing” attempts, which turns out to be effective.

**Phishing Emails and Hosting Servers.** Phishing emails are used to distribute the phishing URLs. Attackers can impersonate trusted parties to send phishing emails via email spoofing [152, 155] or email header injection [244]. In addition to registering squatting domains, attackers can also compromise existing web servers to host the phishing pages [227].

### 6.10 Conclusion

In this chapter, we perform an extensive measurement on squatting phishing, where the phishing pages impersonate target brands at both the domain and content level. By monitoring 700+ brands and 600K squatting domains for a month, we identified 857 phishing web pages and 908 mobile pages. We show that squatting phishing pages are impersonating trusted entities through all different domain squatting techniques. Squatting phishing pages are more likely to adopt evasion techniques and are hard to catch. About 90% of them have evaded the detection of popular blacklists for at least a month.
## 6.10. Conclusion

<table>
<thead>
<tr>
<th>Brand</th>
<th>Squatting Phishing Domains</th>
<th>Squatting Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google</td>
<td>goole.nl</td>
<td>Homograph</td>
</tr>
<tr>
<td></td>
<td>google.pl</td>
<td>Homograph</td>
</tr>
<tr>
<td></td>
<td>googl4.nl</td>
<td>Typo</td>
</tr>
<tr>
<td></td>
<td>google.com.uyl</td>
<td>Typo</td>
</tr>
<tr>
<td></td>
<td>ggoogle.in</td>
<td>Typo</td>
</tr>
<tr>
<td></td>
<td>googlw.it</td>
<td>Bits</td>
</tr>
<tr>
<td></td>
<td>goofle.com.ua</td>
<td>Bits</td>
</tr>
<tr>
<td></td>
<td>goofle.com.ua</td>
<td>Bits</td>
</tr>
<tr>
<td>Facebook</td>
<td>facebooc.com</td>
<td>Homograph</td>
</tr>
<tr>
<td></td>
<td>faceb00k.bid</td>
<td>Homograph</td>
</tr>
<tr>
<td></td>
<td>facebouk.net</td>
<td>Homograph</td>
</tr>
<tr>
<td></td>
<td>facebooook.top</td>
<td>Typo</td>
</tr>
<tr>
<td></td>
<td>face-book.online</td>
<td>Typo</td>
</tr>
<tr>
<td></td>
<td>fakebook.link</td>
<td>Typo</td>
</tr>
<tr>
<td></td>
<td>faebook.ml</td>
<td>Typo</td>
</tr>
<tr>
<td></td>
<td>facebook.ml</td>
<td>Typo</td>
</tr>
<tr>
<td></td>
<td>facecook.mobi</td>
<td>Bits</td>
</tr>
<tr>
<td></td>
<td>facebook-c.com</td>
<td>Combo</td>
</tr>
<tr>
<td>Apple</td>
<td>apple-prizeuk.com</td>
<td>Combo</td>
</tr>
<tr>
<td>Bitcoin</td>
<td>get-bitcoin.com</td>
<td>Combo</td>
</tr>
<tr>
<td>Uber</td>
<td>go-uberfreight.com</td>
<td>Combo</td>
</tr>
<tr>
<td>Youtube</td>
<td>you5ube.com</td>
<td>Typo</td>
</tr>
<tr>
<td>Paypal</td>
<td>paypal-cash.com</td>
<td>Combo</td>
</tr>
<tr>
<td></td>
<td>paypal-learning.com</td>
<td>Combo</td>
</tr>
<tr>
<td>Citi</td>
<td>securemail-citizenslc.com</td>
<td>Combo</td>
</tr>
<tr>
<td>Ebay</td>
<td>ebay-selling.net</td>
<td>Combo</td>
</tr>
<tr>
<td></td>
<td>ebay-auction.eu</td>
<td>Combo</td>
</tr>
<tr>
<td>Microsoft</td>
<td>formateurs-microsoft.com</td>
<td>Combo</td>
</tr>
<tr>
<td></td>
<td>live-microsoftsupport.com</td>
<td>Combo</td>
</tr>
<tr>
<td>Twitter</td>
<td>twitter-gostore.com</td>
<td>Combo</td>
</tr>
<tr>
<td>Dropbox</td>
<td>drapbox.download</td>
<td>Homograph</td>
</tr>
<tr>
<td>ADP</td>
<td>mobile-adp.com</td>
<td>Combo</td>
</tr>
<tr>
<td>Santander</td>
<td>santander-grants.com</td>
<td>Combo</td>
</tr>
</tbody>
</table>

Table 6.14: Selected example phishing domains for 15 different brands. Note that “●” means web page only. “○” means mobile page only. The rest have both web and mobile pages.
Chapter 7

Credential Sharing on Phishing Sites

7.1 Acknowledgement

The leading author of this work is Peng Peng. I am the fourth author of this work. Peng Peng led the entire project and conducted most experiments and data analysis. I helped by setting up an automated eCrimeX query tool to find phishing page links. Then while contacting the phishers, I wrote a tracking service to track the phishers we identified. Finally, I helped setting up proxy for our main phishing experiments.

7.2 Introduction

Phishing attack is a persistent threat on the Internet. It exploits human factors to lure the target users to give away critical information. In recent years, phishing becomes an even bigger concern due to its prevalent usage in facilitating major data breaches [54], particularly the recent breaches in hospitals and health care companies [55, 57]. In addition, phishing plays an important role in many state-sponsored attacks. One of the recent examples is the spear phishing attack against John Podesta, the campaign manager of Hillary Clinton, during the US election in 2016 [49].

The research community has been studying phishing attacks from different aspects. While some existing works analyzed phishing emails [146], the vast majority focus on the phishing websites that are set up by attackers to trick users to reveal important information (e.g., login credentials) [277, 297, 304, 322]. These phishing sites often impersonate other reputable entities to gain the victim’s trust. More recently, researchers analyze phishing kits, the software packages for running phishing websites, to understand how phishing sites are deployed and operated [100, 143, 228]. However, these works only looked into the disconnected parts of phishing. There is a limited end-to-end understanding of the information flow after user credentials are leaked to the phishing sites.

In this chapter, we perform an empirical measurement by piecing together the different stages of phishing to understand the information flow. We collect a large set of live phishing sites and feed fake login credentials to these sites. In this process, we monitor how the information is shared to the attackers who deployed the phishing site, and more importantly, any other
third-parties. For the client-side measurement, we build a measurement tool to automatically detect a login form, fill in the fake credentials, and monitor the network traffic to external parties. For the phishing-server measurement, we build a crawler to retrieve phishing kits, and run them in a sandbox to detect first-party and third-party information collectors. Finally, to examine what attackers do after obtaining the login credentials, we set up our own honey accounts (in email services) to monitor the potential post-phishing exploiting activities. These steps allow us to provide an end-to-end view of the phishing process and credential sharing.

We performed the measurement from August 2018 to January 2019 covering 179,865 phishing URLs. The client-side measurement covers 41,986 live phishing sites, and the server-side measurement is based on the analysis of 2,064 detected phishing kits. Our post-phishing exploitation analysis uses 100 honey accounts from Gmail and 50 accounts from ProtonMail for data collection. We explore how likely attackers would attempt to use the leaked password to further hijack the associated email account (in addition to the original online account).

Our study leads to a number of key findings. First, we show that user credentials are shared in real time on both the client-side and the server-side. This easily exposes the stolen credentials to more malicious parties. Second, while the client-side sharing is not very common (about 5%), the third-party servers are often located in a different country (compared to the phishing server), which may create difficulties to take them down. In particular, many “good” websites were used to receive stolen credentials (e.g., Google Ads are used to track the phishing statistics for attackers). Third, server-side credential sharing is primarily done via emails. 20% of the phishing kits send the credentials to two or more email addresses. About 5% of the phishing kits contain backdoors that stealthily leak the credentials to third-parties. Finally, from our honey email accounts, we observe that attackers indeed attempted to exploit the honey accounts shortly after phishing (within tens of minutes or 1–2 days). A single leakage can attract multiple attackers, which indicates credential sharing.

Our work makes three key contributions:

- **First**, we perform a large-scale empirical measurement on the information flow of credential sharing during phishing attacks. Our measurement covers both client-side, and server-side information sharing, and post-phishing exploitation.

- **Second**, we build a new measurement tool to automatically seed fake credentials to phishing sites to measure the information sharing in real time. We will make the tool available for sharing with the research community.

- **Third**, our measurements provide new insights into the credential sharing mechanisms (to third-parties) during the phishing process.

In the end of the chapter (§7.8), we discuss how third-party sharing and backdoors can be potentially used by defenders for good purposes. For example, the defender may leverage
the third-party sharing channel to establish a vantage point to back-track phishing kit usage, and provide early alerts for phishing victims.

7.3 Motivations

Phishing is an extensively-studied topic, and yet there is still a lack of empirical understanding of the information flow after the credential leakage. Most existing works focus on step 1 to analyze the characteristics of phishing websites and their hosting domains to build detection systems [277, 297, 304, 322]. More recently, researchers analyze the phishing kits to understand how phishing websites are deployed [143, 228]. However, these works are usually limited in scale and scope. More importantly, there is no existing work that systematically measures the real-time credential sharing to third-parties, or examines the post-phishing exploitation activities.

In this chapter, we seek to provide a more comprehensive view of the information flow of phishing attacks via a large-scale empirical measurement. We examine the end-to-end process: from leaking the login credentials to the phishing websites, to analyzing the phishing servers and phishing kits, and monitoring attacker’s exploitation activities using the leaked credentials. More importantly, for the first time, we want to measure the real-time credential sharing to third-party collectors at both client and server sides. Regarding monitoring the account exploitation, the most related work is a study from Google [88] that monitored the activities of manually hijacked Google accounts. Another study leaked email accounts to underground forums and monitored the account activities [230]. These works focus on the generic accounts hijacking, while we specifically focus on the account exploitation after the phishing attack as part of the end-to-end analysis.

7.3.1 Methodology Overview

In this section, we describe our methodology to track the information flow in each step in Figure 2.3. Here, we only describe the high-level idea, and leave the detailed design and analysis to the corresponding sections in the later part of the chapter.

First, to track the information flow at step 1, step 2.1, and particularly step 2.2, we design a measurement tool to automatically feed (fake) login credentials to real-world phishing websites via the login forms. The tool will also keep track any redirections and real-time credential sharing during this process (§7.4 and §7.5).

Second, to infer the information flow of step 3.1, step 3.2, and step 3.3, we try to obtain the phishing kits from phishing servers and analyze how the phishing kits work. We extract the email addresses that first-party attackers use to collect the user credentials. We also perform a dynamic analysis in a sandbox to identify potential backdoors planted by third-parties.
Table 7.1: Dataset summary.

<table>
<thead>
<tr>
<th>Blacklist</th>
<th>Crawling Time Span</th>
<th>Target Brand</th>
<th>Detection Time</th>
<th># All</th>
<th># Live</th>
<th># w/ Login Form</th>
<th># Success</th>
</tr>
</thead>
<tbody>
<tr>
<td>OpenPhish</td>
<td>09/24/2018 - 01/03/2019</td>
<td>✓</td>
<td>✓</td>
<td>75,687</td>
<td>44,553</td>
<td>24,202</td>
<td>19,720</td>
</tr>
<tr>
<td>eCrimeX</td>
<td>08/20/2018 - 01/03/2019</td>
<td>✓</td>
<td>✓</td>
<td>65,465</td>
<td>33,319</td>
<td>21,161</td>
<td>19,172</td>
</tr>
<tr>
<td>PhishTank</td>
<td>09/24/2018 - 01/03/2019</td>
<td>✓</td>
<td>✓</td>
<td>50,608</td>
<td>41,682</td>
<td>7,406</td>
<td>6,430</td>
</tr>
<tr>
<td>PhishBank</td>
<td>09/24/2018 - 01/03/2019</td>
<td>✓</td>
<td>✓</td>
<td>3,093</td>
<td>2,027</td>
<td>1,010</td>
<td>864</td>
</tr>
<tr>
<td>Total</td>
<td>08/20/2018 - 01/03/2019</td>
<td>–</td>
<td>–</td>
<td>179,865</td>
<td>110,934</td>
<td>47,703</td>
<td>41,986</td>
</tr>
</tbody>
</table>

Third, to shed light on step 4, we intentionally leak email addresses and their real passwords via phishing sites, and monitor how attackers would exploit the email accounts after the phishing. These “honey accounts” are created by ourselves and do not affect any real users (§7.7).

7.4 Tool Design & Data Collection

We start by introducing our measurement tool to track the information flow on the client side. Given a phishing website, our tool can automatically detect the login form, fill in the fake credential (email address and password), and submit the information to the phishing server. In this process, our tool records all the HTTP and HTTPS traffic and detect those that transmit the credential to remote servers. In the following, we describe the detailed designs of this tool, and how we collect our datasets.

7.4.1 Measurement Tool

Our tool is a web crawler implemented using Selenium\(^1\). It controls a headless ChromeDriver browser to complete a series of actions and records the network traffic in the ChromeDriver log.

Detecting the Login Form. We focus on phishing sites that collect login credentials, excluding those that collect other information such as credit card information or social security numbers. We detect the login form by looking for three fields: username, password, and the “submit” button. We look for related tags in HTML including FORM tags, INPUT tags and BUTTON tags. We also extract the form attributes such as type, placeholder, name, and class). We don’t consider any read-only or invisible tags.

To make sure that the form is indeed a login form instead of other irrelevant forms (e.g., searching bar, survey forms), we compile a list of login related keywords and search them within the form attributes. We select keywords manually analyzing the login forms of 500

\(^1\)https://www.seleniumhq.org/
randomly phishing websites. In total, we select 40 keywords including 14 keywords for username (e.g., “user name”, “id”, “online id”, “email”, “email address”), 8 keywords for password (e.g., “password”, ”passwd”, “passcode”), and 18 keywords for the submit button (e.g., “log in”, “sign in”, “submit”). The main challenge is that phishing websites often have unconventional designs, or even intentionally hide keywords to evade detection [277]. It is not always possible to locate all three fields. Below, we list the key problems and how to address them.

- **Keywords in images**: The most common challenge is that attackers use an image to contain the “Login” keyword for the submit button, instead of placing the keyword to the placeholder. Our solution is to use the Tesseract Open Source OCR Engine\(^2\) to extract the texts from images, and then perform the keyword search.

- **No FORM tags**: Phishing pages may intentionally leave out the FORM tags (to evade detection). Our solution is to search INPUT tags and keywords in the whole HTML page, instead of just within the FORM tags.

- **Two-step login**: In some phishing pages, users need to enter the username on the first page, and type in the password on the next page. Our tool can handle two-step login by tracking the log-in progress.

- **Previous unseen keywords**: the keywords may occasionally fail to match the corresponding input fields. To increase our success rate, we perform a simple inference based on the order of input fields. For example, if the username and button fields are matched, then we guess the unmatched input field in the middle is for the password.

**Filling in the Fake Credential.** After detecting the login form, our tool will automatically fill in the username and password fields and click the submit button. The username is an email address that belongs to us. The password is a random string of 8 characters which is uniquely created by us. The unique password is helpful later to detect the network requests that send out the password. This email address is never used to register any online account. The password is also not the real password for the email address. In this way, we make sure the leaked information would not affect any real users. We test the tool on 300 phishing sites (different from those that contributed the keywords). We show that the tool has a success rate of 90% to complete the login.

Here, we also want to make sure that using fake credentials does not affect our measurement result. We did a small experiment to see if the phishing site would react to real and fake password differently. We create 4 real accounts with PayPal, Microsoft, LinkedIn, and AT&T respectively. Then we select 60 live phishing websites from eCrimeX that impersonate these brands (15 websites per brand). We feed the real and fake passwords in separate runs, and find that the collected network traffic has no difference.

\(^2\)https://github.com/tesseract-ocr/tesseract
7.4. Tool Design & Data Collection

7.4.2 Data Collection

Using the measurement tool, we collect data from August 2018 to January 2019 by crawling 4 large phishing blacklists: PhishTank, PhishBank, eCrimeX, and OpenPhish. The detailed data statistics are shown in Table 7.1. For each phishing URL, all four blacklists share the timestamp when the phishing URL was reported/detected. Three of the blacklists also show the target brand (or website) that the phishing page is trying to impersonate. OpenPhish shares the target brand information only for the premium API (not the free-API we used). We notice that many phishing URLs become inaccessible quickly after they are blacklisted. To interact with the live phishing server, we build a crawler to fetch phishing URLs from the four blacklists every 30 minutes. Then we immediately use our measurement tool to load the phishing page, feed the fake credential, and record the network traffic.

We also considered that situation where the phishing servers use cloaking techniques. More specifically, the phishing server may check the IP and User-Agent of the incoming request to see if the request is coming from a university, a security company, or a web crawler. In those cases, the phishing server may drop the request or return a benign page to avoid being detected. As such, we put our crawler behind web proxies and use a realistic User-Agent.

As shown in Table 7.1, we collected 190,087 unique phishing URLs (after removing duplicated URLs between the four blacklists). Among them, 68,751 (38.26%) are “dead”, and the rest 110,934 (61.74%) are still alive. Figure 7.1 shows that the live pages are typically more recently-reported compared to the dead ones. 80% of the live pages were reported just 1 hour ago (by the time we visited the pages), while the dead pages were reported much earlier.

Login Results. Not all the live URLs are still phishing pages. In fact, many of the live URLs have been reset to legitimate/blank pages. Among 110,934 (61.74%) live URLs, only 47,703 (26.55%) still contain a login form. We use our measurement tool to feed the
Hash or encoding functions (31 in total)

| MD2, MD4, MD5, RIPEMD, SHA1, SHA224, SHA256, SHA384, SHA512, SHA3_224, SHA3_256, SHA3_384, SHA3_512, blake2b, blake2s, crc32, adler32, murmurhash 3 32 bit, murmurhash 3 64 bit, murmurhash 3 128 bit, whirlpool, b16 encoding, b32 encoding, b64 encoding, b85 encoding, url encoding, gzip, zlib, bz2, yenc, entity |

Table 7.2: Functions used to obfuscate login credentials.

(a) acm.org  
(b) sigcomm.org  
(c) nsw.gov.au

Figure 7.2: Compromised domains and their hosted phishing pages.

fake credentials to and record all the network traffic. Out of the 47,703 phishing sites, we successfully submitted the login form for 41,986 sites (88.01%). We manually checked the pages with failed logins. Some of the forms not only asked for username and password, but also required answering security questions by clicking a drop-down list. Other failure cases are caused by the special format constraints for the input data. We admit that there is still room for improving our measurement tool.

Identifying Relevant Network Traffic. Among all the network requests, we look for those that contain the seeded password. We consider both POST and GET HTTP/HTTPS requests. We expect that some phishing pages may encode or hash the credentials before transmission. As such, in addition to matching the plaintext, we also attempt to match the hashed/encoded versions of the password. We apply 31 hash/encoding function on the password and look for a match in the traffic (Table 7.2). After the filtering, we identified 41,986 network requests that contain the leaked password (either plaintext or hashed).

7.5 Client Side Analysis

We now analyze the collected dataset to examine the various aspects of the phishing websites including their target brands, domains and server geolocations. Then we inspect the information flow to understand how the login credentials are shared with third-party information collectors. The analysis of this section is based on the 47,703 phishing sites with a login form.
7.5. Client Side Analysis

<table>
<thead>
<tr>
<th>Rk.</th>
<th>Domain Name</th>
<th># Unique URLs</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>kylelierman.com</td>
<td>3,257 (6.82%)</td>
<td>Uncategorized</td>
</tr>
<tr>
<td>2</td>
<td>datarescue.cl</td>
<td>545 (1.14%)</td>
<td>Phishing &amp; frauds</td>
</tr>
<tr>
<td>3</td>
<td>psycheforce.com</td>
<td>519 (1.09%)</td>
<td>Sex Education</td>
</tr>
<tr>
<td>4</td>
<td>4-6-3baseball.com</td>
<td>447 (0.94%)</td>
<td>Web Hosting</td>
</tr>
<tr>
<td>5</td>
<td>serveirc.com</td>
<td>424 (0.89%)</td>
<td>Dynamic DNS</td>
</tr>
<tr>
<td>6</td>
<td>galton.pila.pl</td>
<td>303 (0.63%)</td>
<td>Retail and Wholesale</td>
</tr>
<tr>
<td>7</td>
<td>lexvidhi.com</td>
<td>287 (0.60%)</td>
<td>Business Marketing</td>
</tr>
<tr>
<td>8</td>
<td>xsitedleadpages.com</td>
<td>262 (0.55%)</td>
<td>Uncategorized</td>
</tr>
<tr>
<td>9</td>
<td>stcroixlofts.com</td>
<td>233 (0.49%)</td>
<td>Dynamic Content</td>
</tr>
<tr>
<td>10</td>
<td>colorsplashstudio.com</td>
<td>230 (0.48%)</td>
<td>Blogs &amp; shopping</td>
</tr>
</tbody>
</table>

Table 7.3: Top 10 domains of phishing URLs.

7.5.1 Understanding Phishing Sites

HTTPS Scheme. HTTPS is already widely used by the phishing sites. Among the 47,703 sites, 16,128 (33.81%) are hosting the phishing pages under HTTPS. We suspect that HTTPS helps to further deceive the users. More specifically, most modern browsers display a green padlock as the security indicator for HTTPS sites (with a valid certificate). This means, if a phishing site enables HTTPS, the green padlock would also show up when a user visits it. This could give the user a false sense of “security” given that user may not fully understand the meaning of the security indicator [123].

Domain Analysis. The 47,703 phishing sites are hosted under 24,199 full qualified domain names (FQDNs) which correspond to 16,939 unique domain names. Table 7.3 shows the top 10 domains ranked by the number of unique phishing URLs. There is no single domain that has a dominating contribution to the phishing URLs.

Interestingly, 417 domains are ranked within Alexa top 1 million³. We then manually investigate those domains, and classify them into four categories: 159 domains belong to web hosting services, 3 domains belong to dynamic DNS services, and 31 domains belong to URL shortener services. The rest 224 domains can not be easily categorized since they look like good websites that got compromised. In Table 7.4, we list the top 10 domains (based on their Alexa ranking) that are likely compromised for phishing.

Figure 7.2 shows three examples of compromised websites. Figure 7.2a is a phishing page hosted under acm.org. The phishing URL is “http://iccps.acm.org/admin/.certified/***” deployed under the ICCPS conference site to impersonate the FedEx website. Figure 7.2b is a phishing URL ”http://conferences.sigcomm.org/css/***” hosted under the SIGCOMM conference website to impersonate a tax agency in France. Figure 7.2c is a phishing URL hosted under a government website of New South Wales in

³https://www.alexa.com/topsites
Chapter 7. Credential Sharing on Phishing Sites

<table>
<thead>
<tr>
<th>Domain</th>
<th>Alexa rank</th>
<th># URLs</th>
</tr>
</thead>
<tbody>
<tr>
<td>archive.org</td>
<td>269</td>
<td>1</td>
</tr>
<tr>
<td>bathandbodyworks.com</td>
<td>1,224</td>
<td>4</td>
</tr>
<tr>
<td>etherscan.io</td>
<td>3,162</td>
<td>3</td>
</tr>
<tr>
<td>nsw.gov.au</td>
<td>3,182</td>
<td>1</td>
</tr>
<tr>
<td>acm.org</td>
<td>3,676</td>
<td>11</td>
</tr>
<tr>
<td>tillys.com</td>
<td>9,506</td>
<td>1</td>
</tr>
<tr>
<td>krakow.pl</td>
<td>10,902</td>
<td>5</td>
</tr>
<tr>
<td>ugm.ac.id</td>
<td>11,198</td>
<td>1</td>
</tr>
<tr>
<td>kemkes.go.id</td>
<td>12385</td>
<td>4</td>
</tr>
<tr>
<td>mun.ca</td>
<td>13036</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 7.4: Compromised domains that host phishing pages.

Figure 7.3: Geolocation distribution of phishing URLs.

Geolocation Analysis. We further examine the geolocation of the phishing servers\(^4\). In this analysis, we do not consider phishing pages under web hosting services or compromised domains since these servers are not dedicated phishing servers. In total, we have 10,192 unique IP addresses, and their geolocation distribution is shown in Figure 7.3. The majority of the phishing sites are hosted in North America and Europe, especially in the United States. This result, in part, can be biased due to the fact that the phishing URLs are collected from four US-based phishing blacklists.

Target Brands. The phishing sites are impersonating a wide range of popular “brands”. Recall that three of the four blacklists provide the target brand information, which covers 28,614 URLs (59.99% out of 47,703). For the rest 19,089 phishing URLs, we need to identify the target brands by ourselves. Our method is based on those in \([304, 322]\). The intuition is that a target brand that the phishing website is impersonating is typically more popular

\(^4\)For geolocation service, we use the GeoPlugin (https://www.geoplugin.com/).
7.5. Client Side Analysis

<table>
<thead>
<tr>
<th>Target Sectors</th>
<th># Phishing Sites</th>
<th># Brand</th>
<th>1st brand</th>
<th>2nd brand</th>
<th>3rd brand</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finance and Insurance</td>
<td>18,648 (39.09%)</td>
<td>150</td>
<td>PayPal (15,083)</td>
<td>Desjardins (960)</td>
<td>Wells Fargo (646)</td>
</tr>
<tr>
<td>Computer and Software</td>
<td>9,304 (19.50%)</td>
<td>58</td>
<td>Microsoft (4,484)</td>
<td>LinkedIn (761)</td>
<td>Yahoo (603)</td>
</tr>
<tr>
<td>Electronic and Comm.</td>
<td>1,262 (2.65%)</td>
<td>23</td>
<td>AT&amp;T (927)</td>
<td>Apple (161)</td>
<td>Verizon (29)</td>
</tr>
<tr>
<td>Transportation Services</td>
<td>583 (1.22%)</td>
<td>9</td>
<td>Federal Express (393)</td>
<td>DHL (13)</td>
<td>Delta (40)</td>
</tr>
<tr>
<td>Other</td>
<td>5,456 (11.44%)</td>
<td>48</td>
<td>eBay (159)</td>
<td>Craigslist (126)</td>
<td>IRS (124)</td>
</tr>
<tr>
<td>Not Applicable</td>
<td>12,450 (26.10%)</td>
<td>10</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 7.5: Target sectors and top brands in each sector.

(i.e., ranked higher in the search engine). For each of the 19,089 phishing pages, we first apply the OCR technique [168] to extract keywords from the webpage screenshot. Here, we use screenshots instead of the HTML file because attackers often use obfuscation techniques to hide the keywords in HTML [277]. Then we use RAKE (Rapid Automatic Keyword Extraction) [253] to extract keywords from the texts to remove less important keywords (e.g., stop-words). We search the keywords using Google, and take the first returning page as the target brand. For example, if we search the keywords in Figure 7.2c, Google will return paypal.com as the first return result (i.e., the target brand).

We evaluate this approach using phishing pages with known target brands. We first test the method on 500 phishing pages that impersonate Paypal, and get a 100% accuracy. Then we test the method on 500 phishing pages targeting Microsoft, and get a 99.8% accuracy. Finally, we test the method on 500 randomly phishing pages, which returns an accuracy of 88%. We believe this is good enough to proceed with our analysis.

In total, we find 298 unique target brands. The most popular target brand is Paypal, followed by Microsoft, AT&T, Desjardins, and Linkedin. We further categorize the target brands into 6 sectors based on their Standard Industrial Classification (SIC) code. We get SIC code information from siccode.com. As shown in Table 7.5, more than 40% of phishing URLs are targeting finance and insurance services. Paypal alone is associated with 15,083 phishing URLs (32%). Note that 12,450 (26%) phishing sites don’t have an informative target brand. For example, the blacklist may label them as “Generic” or “United States”. Manual inspection reveals that these phishing sites are impersonating small organizations.

### 7.5.2 Client-Side Information Flow

In this section, we investigate the information flows of sending credentials from the client side. To identify HTTP requests containing user credentials, we follow the methodology discussed earlier in §7.4.2. Out of the 47,703 phishing sites with a login form, we are able to track credential information flow for 41,986 phishing sites.

**Credential Sending Format.** Recall that credential information could be transmitted in plaintext or using some encoding/hashing schemes (e.g., MD5, SHA256). Table 7.6 shows statistics of different types of data formats used across phishing sites. Interestingly, most phishing sites (99%) use human interpretable formats (i.e., either plaintext or URL encod-
Chapter 7. Credential Sharing on Phishing Sites

<table>
<thead>
<tr>
<th>Format</th>
<th>Plaintext</th>
<th>URL Encoding</th>
<th>Other Encoding</th>
</tr>
</thead>
<tbody>
<tr>
<td># Phishing sites</td>
<td>6,324 (15.06%)</td>
<td>35,616 (84.83%)</td>
<td>46 (0.11%)</td>
</tr>
</tbody>
</table>

Table 7.6: Data format of credentials sent from the client-side.

<table>
<thead>
<tr>
<th># 3rd-parties</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>≥ 3</th>
</tr>
</thead>
<tbody>
<tr>
<td># Phish sites</td>
<td>39,967 (95.19%)</td>
<td>1,963 (4.68%)</td>
<td>48 (0.11%)</td>
<td>8 (0.02%)</td>
</tr>
</tbody>
</table>

Table 7.7: Distribution of third-party collectors. About 95% phishing sites don’t have third-party collectors and they only send credentials to the original hosting domain.

Identifying Third-party Collectors. Any domain that collects credential information, and is not a direct phishing server domain, is considered to be a third-party collector. In total, we identify 694 third-party collector domains that include 1,021 URLs. These are entities that collect stolen credentials, and would be a vital component to target while building phishing defenses.

But do all phishing sites share credentials with third-party collectors? Table 7.7 shows the distribution of phishing sites that share credentials with different number of third-party collectors. There are about 5% of phishing sites sharing credentials with third-party collectors from the client side. The percentage is not high, but there is a sizeable number. There are 2,019 phishing sites that interact with one or more third-party collectors. In fact, 56 phishing sites share with more than 2 third-party collectors.

Third-party Collectors vs. Phishing Sites. Next, we look at two aspects of third-party collectors that have implications for disrupting their network. First, do third-party collectors link with multiple phishing sites? If each third-party collector served a single phishing site, we would have to take down as many collector domains as the number of phishing sites. But we observe a different trend. Figure 7.5 shows the distribution of fraction of phishing sites covered by different external collectors. We find that the top 100 external collectors (out of 694) link with a majority, 68.76% of the phishing sites. Thus, even targeting a small fraction of external collectors can disrupt many phishing efforts.

Second, we further examine the geographical locations of third-party collectors. Third-party collectors are spread over 37 countries, but 42% of them are located in the U.S. When third-party collectors are based in a country different from the phishing site they link with, it would require different law enforcement efforts to take down their domains. We analyze the relative locations of phishing sites and their associated third-party collectors. Among 1,408 IP address pairs made of phishing sites, and their connected collector domains, 44% are co-

\(^5\)In total, there were 2,170 pairs, but we were unable to determine the geolocation for all of them.
located in the same country. A significant fraction of this number can be attributed to the U.S.—96% of co-located pairs are located within the U.S. The remaining 56% non-co-located pairs include phishing sites that are spread over 52 countries, and collectors over 37 countries. We also note that a significant fraction, 88% of non-co-located pairs involve phishing sites or collectors based in the U.S. The detailed breakdown for is shown in Figure 7.4. We only show the top 5 countries of phishing servers and third-party collectors and group the rest into “other”. Overall, this means that a majority of pairs are not based in the same country, and this could raise challenges to disrupt their network.

**How Reputed are Third-Party Collectors?** We investigate whether the third-party collectors are already known malicious entities or those with poor reputation.

We start by analyzing the reputation of third-party collector domains using *The Talos IP and Domain Reputation Center (by Cisco)*\(^6\). The Talos IP and Domain Reputation Center is a real-time threat detection network. They provide a reputation score of “Good”, “Neutral” and “Poor”. Here “Good” means little or no threat activity has been observed. On the contrary, “Poor” indicates a problematic level of threat activity has been observed, while “Neutral” means the domain is within acceptable parameters. Note that “Neutral” is a common case for most domains, even well-known ones such as facebook.com. Among all 694 third-party collector domains, we obtain reports for 508 (73.20%) domains. We find that 14 of them are labeled “Good”, 146 are “Poor” and the rest 348 are “Neutral”.

We take a closer look at these scores—First, it is interesting to see that a significant fraction, 29% of domains already have poor reputation, but still managed to stay alive and form

---

\(^6\)urlhttps://www.talosintelligence.com/
a collector network. Second, it is surprising to see 14 domains marked as “Good”. We find these are indeed legitimate domains, e.g., delta.com, google.com, doubleclick.net, dropbox.com. On examining the HTTP logs for these “Good” collectors, we find there are different reasons for them acting as third-party collectors. For example, certain phishing sites were sending the credentials to the legitimate sites that they were trying to impersonate (e.g., delta.com). We suspect that they were trying to check the validity of credentials. Some good sites were collecting credentials because they were used by attackers as a web hosting service (e.g., dropbox.com). Finally, popular ads platforms or tracking services such as Google Ads and doubleclick.net also received the stolen credentials. A close inspection shows that the phishing sites were connecting to these tracking services to keep track of the number of victims. While doing so, the stolen credential was “accidentally” placed within the referer URL of the request.

Only analyzing domain reputation does not provide the full picture. There can be legitimate domains that host malicious URLs. We leverage VirusTotal\textsuperscript{7} to scan external collector URLs. VirusTotal has been widely used by the security community in prior work [209, 277]. For each submitted URL, VirusTotal provides a report from 66 diverse scanners that may classify it into one or more categories that indicate whether a URL is problematic, clean or unrated.

\textsuperscript{7}https://www.virustotal.com

---

Figure 7.5: Distribution of fraction of phishing sites that connect to different third-party collectors. On the x-axis, third-party collectors are ranked based on # of phishing sites connected.

Figure 7.6: CCDF of Number of VirusTotal scanners that flagged the given URL as malicious. The majority of the third-party collectors are already flagged by VirusTotal scanners.

Figure 7.7: Registration time of phishing domains and third-party collector domains. Third-party collector domains have a similar distribution with phishing domains.

<table>
<thead>
<tr>
<th></th>
<th># Phishing Sites w/ Third-party Collectors</th>
<th># Third-party Collector URLs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total</td>
<td>2,019</td>
<td>1,021</td>
</tr>
<tr>
<td>“Phishing Site”</td>
<td>1970 (97.57%)</td>
<td>823 (80.63%)</td>
</tr>
<tr>
<td>“Malicious Site”</td>
<td>1,840 (91.13%)</td>
<td>777 (76.10%)</td>
</tr>
<tr>
<td>“Malware Site”</td>
<td>239 (13.13%)</td>
<td>176 (17.24%)</td>
</tr>
</tbody>
</table>

Table 7.8: Number of URLs detected by VirusTotal.
Problematic categories include “Malware site”, “Phishing site”, “Malicious site”, “Suspicious site”, and “Spam site”.

Figure 7.6 shows the distribution of collector URLs detected by VirusTotal scanners that fall into any one of the problematic categories. A small fraction, 16% of URLs are not flagged by any scanner, and will likely remain under the radar for a long time. On the other hand, a large majority, 84% of collector URLs are classified as problematic by at least one scanner. Table 7.8 shows a further breakdown of collector URLs that are flagged by at least one scanner. Interestingly, 81% of them are flagged as ‘Phishing sites’. This suggests the possibility of a network of phishing sites that exchange credential information with each other.

To summarize, while a majority of third-party collector domains do not have a poor reputation, a large majority of their URLs are already known to be problematic, e.g., for phishing. In spite of the poor URL reputation, it is surprising that these collector URLs are still alive. To understand the age of the collector domains, we examine WHOIS records to determine their domain registration dates. Figure 7.7 shows that the distribution of domain registration time of third-party collectors is quite close to that of the phishing servers. Many of the collector domains are actually aged domains. 20% of them were registered 10 years ago. About half of them were registered before 2016. This suggests that the collector network has largely remained undisrupted.\(^8\)

The top information collectors ranked by the number of phishing sites they serve is presented in Table 7.9. The largest information collectors here is “w32.info”. This site was once hosting many phishing kits for downloading (not anymore). We confirm this by check-

\(^8\)We removed known web hosting domains (as reported by Alexa top 1 Million) from this plot to avoid a possible wrong interpretation. Malicious collector URLs hosted on a legitimate webhosting service would show up as being long-lived, while the exact age of the URL would be hard to determine.
ing the achieved versions of this website\(^9\). It is possible that the kit developers were using this site to collect a copy of the stolen credentials from people who use their kits to perform phishing. We also notice that web hosting services or dynamic DNS services are often used to collect credentials for multiple collector URLs (possibly for different attackers). One interesting case is \texttt{ip-api.org}, a website that provides a lookup service for IP geolocations. 89 phishing websites were sending stolen credentials to this server via “\texttt{http://cdn.images.ip-api.org/s.png}”. We suspect that this service might have been compromised.

### 7.6 Server side analysis

In this section, we move to the server side to analyze the information flow of credential transmission. The challenge here is that we don’t have internal access to the phishing servers. Our solution is based on the fact that some (careless) attackers may have left the phishing kit in publicly accessible locations on the phishing server\(^{10}\). As such, we attempt to retrieve these phishing kits and infer the server-side information flow by combining static and dynamic analysis.

#### 7.6.1 Collecting Phishing Kits

We search for phishing kits on servers that host the phishing websites. Unlike §7.5, we inspect all 179,865 phishing URLs (\textit{i.e.,} not just sites that were still alive) for possible phishing kits. The main reason is that even if a phishing site has been disabled\(^{10}\), it is possible that phishing kits are still left accessible on the server\(^{228}\).

Since we have no knowledge of possible file names to query for (on the phishing server), we start with phishing servers that enable directory listing to obtain a list of files available on the server. Prior work suggests that phishing kits are usually compressed/archive files (\textit{e.g.,} zip, tar, rar)\(^{100}\). For each phishing site URL, we do the following steps: (1) Check if directory listing is available for each path segment in the URL (\textit{i.e.,} separated by `/`). (2) If we find a directory listing, we download all compressed/archive files. (3) For each downloaded file, we decompress it and check the PHP/Python/Ruby/HTML files to make sure it is indeed a phishing kit. To further increase our chance to retrieve more phishing kits, we identify the most frequent 50 kit names (based on the first 1000 kits downloaded earlier). Then given a phishing URL, we exhaustively query each path segment for these 50 file names, in addition to checking the directory listing. This helps us to obtain kits from servers that disabled the directory listing.


\(^{10}\)By disabled we mean the phishing site has been reset to a legitimate website by phisher or the web administrator.
We applied the above method to querying 179,865 phishing sites, and obtained 2,064 phishing kits in total. Compared to earlier work [52, 143], our hit rate for finding a phishing kit on phishing servers is lower—we observe a hit rate of 1.15%, compared to 11.8% in prior work. We suspect that phishers are being more careful, and avoid leaving publicly visible traces of their malicious activity.

### 7.6.2 Server-side Information Flow

Unlike client-side analysis, where we only investigate outgoing HTTP/HTTPS requests, information flow on the server side can use other channels too—via Email [143]. Our goal is to capture the information flow on the server side, and also detect those related to third-party credential sharing.

**Identifying Third-party Collectors.** On the server side, the stolen credentials can be sent to third-parties in addition to the attacker who deployed the phishing kit. More specifically, prior work shows that phishing kits may contain backdoors [100] that allow third-parties to collect the stolen credentials. Often cases, the backdoors are stealthily inserted into the phishing kit code by the kit developers. When the kit is used by attackers to perform phishing, the kit developer also receives a copy of the credentials.

To differentiate backdoor collectors, we conduct both dynamic and static analysis. The methodology is inspired by that in [100]. The assumption is that backdoors are usually planted stealthily, which are not directly visible in plaintext in the kit code. As such, we first apply static analysis by performing a text search within files in a kit to identify email addresses, and URL endpoints (for HTTP requests) that collect credentials. Then we put the phishing kit in a sandbox for a dynamic analysis to capture all the outbound HTTP and email traffic that transmit the stolen credentials. Any collector identified from dynamic analysis, but not identifiable via plain text search through static analysis, can be considered to be a backdoor collector (i.e., the third-party). Note that throughout our dynamic analysis, we did not observe any outbound HTTP/HTTPS traffic from any phishing kits. For brevity, we only introduce the details of the email channel analysis below.

**Static and Dynamic Analysis.** Our static analysis is based on a simple method to extract the collectors in plaintext. The idea is to locate the `mail(to,subject,...,header)` function and identify their “to” and “header” variables. The “to” address is considered to be a collector on the server side. Out of 2,064 phishing kits in total, we successfully detected email addresses in 1,974 phishing kits. In total, we extracted 1,222 valid email addresses (as receivers).

For the dynamic analysis, we build up an Apache web server and upload all phishing kits to it. We record all the outbound traffic but block the corresponding ports (e.g., port 25 for email) to avoid actually sending data to the attackers. For each phishing kit, since we do not know which files build the phishing pages, we run our tool described in §7.4.1 to
detect login forms to locate the phishing page. Then like before, we use our measurement tool to automatically fill in the username and password, and submit the information to the experimental server. To capture the server-side actions, we dump all the emails in the mail queue and all the HTTP logs.

We run the dynamic analysis on all of the 2,064 phishing kits. Using tools described in §7.4.1, we successfully logged into 1,181 (57%) phishing kits. Note that for 88 (9%) of these phishing kits, we did not find any outbound emails. It is possible that these attackers would rather log into the phishing server to retrieve the stolen credentials (step 3.2 in Figure 2.3). For the rest of the phishing kits, we search the leaked password in their outbound emails to make sure they are sending the stolen credentials. We only find 6 emails that did not contain the password (the emails were for status reports). For these 1,093 phishing kits, we compare the result of dynamic analysis and that of static analysis, and find 46 phishing kits with backdoor emails (4.2%).

Server-side Collectors. Figure 7.8 shows the number of server-side collectors per phishing kit. Each collector is identified as a receiver email address. Most phishing kits (96%) do not have a backdoor (third-party) collector. Among the 46 kits that have a backdoor, there is usually only one backdoor collector per kit. In total, there are 24 unique backdoor email addresses. Table 7.10 further displayed the top 5 third-party email addresses, ranked by the number of associated phishing kits. Some collectors (e.g., equallib12@gmail.com) were embedded into multiple phishing kits.

Regarding the first-party collectors, Figure 7.8 shows that most phishing kits have one first-party collector, but about 20% kits have more than one collectors. As shown in Table 7.11, some of the first-party collectors are associated with multiple kits, which indicates coordinated phishing campaigns, i.e., one attacker deployed the kits onto multiple phishing servers.

Comparing Client-side and Server-side Flows. We next compare the flows of
the client side with those of the server side. Among 179,865 phishing URLs, we find 2,064 phishing kits from 1,286 phishing server domains. 437 (34.0%) of these phishing domains overlap with those live phishing sites analyzed in §7.5. Given a phishing server domain, we examine the number of client-side collectors and the number of server-side collectors (combining first- and third-parties). The results are shown in Table 7.12. The majority of the domains (296 domains, 67.7%) has one collector on the server-side without any client-side collector. Only a small number of domains (7 domains, 1.6%) have collectors at both sides. There are 18 domains that have no collectors at neither sides. In this case, attackers would need to login to the phishing server to collect the stolen credentials.

## 7.7 Post-Phishing Exploitation

So far we explored different ways of information leakage, but what would the attackers do with the stolen credentials? To shed light on the post-phishing exploitation activities, we set up honeypot accounts whose credentials are intentionally leaked by us to phishing sites. Then by developing a honeypot account monitoring system, we can uncover activities that access and use our honeypot accounts. This idea is inspired by prior work by Onaolapo et al. on monitoring the activities after account hijacking [230]. While Onaolapo et al. investigated account hijacking more broadly, our focus is specifically on the fate of accounts that have credentials stolen by phishing attacks. This analysis helps to complete a comprehensive end-to-end view of what happens after the information leakage.
Table 7.12: Collectors on both client and server side.

<table>
<thead>
<tr>
<th>Id</th>
<th>Honey Account</th>
<th>CTRY</th>
<th>Target Brand</th>
<th>Leak Time</th>
<th>First Login Time</th>
<th>#Login (#IP)</th>
<th>Login CTRY</th>
<th>#Email Read</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Protonmail</td>
<td>DE</td>
<td>Generic Email</td>
<td>10/26/2018 15:27</td>
<td>10/26/2018 16:50</td>
<td>7 (4)</td>
<td>NG</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>Protonmail</td>
<td>US</td>
<td>LinkedIn</td>
<td>10/26/2018 15:21</td>
<td>10/26/2018 20:15</td>
<td>6 (4)</td>
<td>NG, CN</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>Protonmail</td>
<td>US</td>
<td>LinkedIn</td>
<td>10/26/2018 15:20</td>
<td>10/28/2018 14:14</td>
<td>1 (1)</td>
<td>GH</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>Protonmail</td>
<td>US</td>
<td>Microsoft</td>
<td>12/21/2018 1:08</td>
<td>12/22/2018 1:46</td>
<td>1 (1)</td>
<td>PK</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 7.13: Account exploitation activities in our honey accounts.

### 7.7.1 Experiment Setup

Our goal is to understand the post-phishing exploitation on the email accounts. For example, suppose the attacker sets up a phishing site to impersonate “paypal.com” to steal PayPal accounts, we expect the attacker will first try to login to the PayPal account (e.g., to steal money). As the second-step exploitation, the attacker may also try to hijack the email account that is associated to the PayPal account using the same password (assuming users reuse the password). Intuitively, the email account can be used to hijack other online accounts registered under this email (e.g., through password reset), and thus has value. In the following, we set up honey email accounts to study this second-step exploitation.

**Honeypot Accounts Setup.** Our honeypot accounts include two different types of email accounts: Gmail and ProtonMail\(^{11}\). Gmail is a large popular email service provided

\(^{11}\text{https://protonmail.com/}\)
by Google, while ProtonMail is a less popular end-to-end encrypted email service based in Switzerland. We manually created 100 Gmail and 50 ProtonMail accounts and assigned them random combinations of popular first and last names. To make the freshly-created email accounts believable and realistic, we populated them with emails from the public Enron email dataset [175]. Enron dataset contains emails sent by executives of the energy corporation Enron, and was publicly released as evidence for the bankruptcy trial of the company. To avoid causing suspicion from attackers, we applied the following method to modify those emails before putting them into the inbox of the honey accounts. First, we translated the old Enron email timestamps to recent timestamps slightly earlier than our experiment start date. Second, we replaced the sender domain with some popular email domain such as gmail.com and outlook.com. Third, we replaced all instances of “Enron” with a fictitious company name.

For all the honey accounts, we did not enable any type of two-factor authentications. This is to make sure the attackers can perform the login using username and password alone. We also perform a quick confirmation test. We attempted to log in to these honey accounts from different countries (using web proxies), and found that the logins were all successful.

**Leaking Real Credentials.** To leak the credentials of the honey accounts, we choose phishing sites from 4 categories based on their target brands: “PayPal”, “Finance and Insurance”, “Computer and Software Services”, and “Others”. We treat PayPal as a separate category since a major portion of the phishing sites target the PayPal brand (see Table 7.5). Phishing sites that target “Electronic and Communication” and “Transportation Services”, account for less than 10% of our data, so we count them as “Others”. We choose 150 phishing sites (roughly 40 phishing sites from each category), and leak one email credential to each site (thus using all our honeypot accounts). The freshly created honey account is exclusively leaked to one phishing site only, which helps us to accurately attribute the exploitation activities to the original phishing site.

**Monitoring Infrastructure.** We develop our own monitoring system to collect data about the account activities. For Gmail, we obtain the information of recent logins from the “last account activity” page. Each login record contains the IP, device information, and timestamp of login. Similarly, ProtonMail also provides such logs in its security settings. For both providers, we develop a script that can automatically login to each account and crawl the information of recent login records. To further monitor attacker activities after login, we obtain the scripts used in [230] to scan the inbox and detect any changes. The activity logs are periodically sent to a separate email account (created for data collection) under our control.

**Ethical Considerations.** The above experiment requires ethical considerations. First, all the honey accounts are freshly created by us, and the experiment would not affect any real users of Gmail or ProtonMail. Second, to run this experiment, we need to give attackers

\[\text{https://support.google.com/mail/answer/45938?hl=en}\]
the access to honey accounts. A potential risk is the attackers may use the honey accounts for other malicious activities. To reduce the risk, we restrict our ourselves to a small-scale experiment. This means attackers do not get many accounts. In addition, all the historical emails and previous contacts in these accounts are synthetically created. This means attackers cannot use these honey accounts to further phish their contacts (a common way of performing spear phishing). Throughout the experiment, these honey accounts are never used to send any emails. Third, we make sure to delete the honey accounts after the experiment.

### 7.7.2 Activities on Honeypot Accounts

Starting in November 2018, we performed the experiment by manually leaking the honey account credentials (email address and password) to different phishing sites. The credentials were not all leaked at once. After the credentials were leaked, we monitored the honey account for at least 50 days. Out of the 150 honey accounts, we observe that 7 accounts (leaked to different phishing sites) have received logins. Table 7.13 summarizes the account activities.

**Overall Observations.** First, we observe that the exploitation happened very quickly after the credential leakage. It can be shortly within an hour or only after 1–2 days. Second, most of the times, the attackers logged in from countries different from where the original phishing sites were located. Third, for some honey accounts, there are often multiple login attempts from different IP addresses. The result echoes our early analysis that the stolen credentials can be leaked or shared to multiple attackers.

**Detailed Activity Analysis.** Next, we provide more detailed results for each of the honey accounts.

- **Account-1** is the only Gmail account that received logins. The original phishing site is hosted in Arizona, US. After 35 minutes of the credential leakage, attackers first logged in from Boston, US. After that, the attacker registered an Amazon Web Service (AWS) account using the honey account which left a confirmation email in the inbox. A few minutes later, the honey account received an email that indicated AWS payment failure. In the following 5 days, the attacker kept logging into the account for 8 additional times from the same IP address, but did not have other observable activities.

- **Account-2, 5 and 6** has one login each. All three phishing sites are hosted in the U.S., but all the logins are originated from a different country—Morocco, Ghana, and Pakistan. In addition, in Account-2, 5, and 6, the attacker read 1, 2, and 2 emails each, respectively. We suspect they were searching for something of value in the account, e.g., banking information, social security numbers, credentials to other services.
• **Account-3** has 7 logins using 4 IPs from Nigeria, despite the phishing site being hosted in France. We did not observe any patterns in account access; they did not check the account on consecutive days.

• **Account-4** is more interesting as we observe activities from 2 different countries. After about 5 hours of the leakage, the attacker first logged in from Nigeria. Then 3 days later, we saw two logins from Beijing, China. Half a month later, the first attacker from Nigeria (i.e., using the same IP) checked the account again. This phishing site is also hosted in the US. It is possible that the credential is leaked to multiple attackers during phishing\textsuperscript{13}. The attackers read 4 emails.

• **Account-7** is another one with login activities from different locations—5 different cities (3 countries). There are 17 different logins over a period of one month. First, the attacker logged in from Lagos, Nigeria. Two days later, another attacker logged in from Atlanta, US. And then, on Jan 3, 2019, there were two logins from Burnaby, Canada and one from Miami, US. The last login was found from Los Angeles, US. We believe this could be evidence for credential sharing. Also, 1 email was read.

From our analysis, we conclude that attackers indeed log in to the email accounts and check whether they can find anything of value (by reading emails). Recall that the email accounts were not the initial targets of the phishing attack—the initial targets were online accounts of PayPal, LinkedIn, Microsoft. This explains why only 5% of attackers would go the extra miles to the hijacking of the associated email accounts. The irregular patterns of the account activities also suggest that the exploitation is likely done manually.

\textbf{7.8 Discussion}

\textbf{Implications of Results.} Our measurement results have several key implications. \textit{First}, credentials sharing happens throughout the phishing process at both client and server side, which exposes the stolen credentials to more malicious parties. The good news is that third-party sharing is not yet prevalent. \textit{Second}, from the phisher’s perspective, credential sharing can be both intended (\textit{e.g.}, for validating the stolen credentials and tracking attack statistics) or unintended (\textit{e.g.}, due to backdoors planted by phishing kit developers). \textit{Third}, from the defender’s perspective, client-side phishing efforts are easier to detect. In §\textbf{7.5}, we find that over 80\% of client-side 3rd-party collectors are already flagged by VirusTotal. However, the problem is that they were not effectively taken down (they are usually in a different country compared to the phishing site). Nevertheless, defense schemes can still add these domains into local network blacklists to block credential sharing. \textit{Fourth}, server-side efforts are harder to measure and disrupt. Web-hosting platforms can significantly contribute

\textsuperscript{13}We cannot confirm whether there was server-side sharing since the phishing kit was not accessible. We did not observe any client-side sharing on this phishing site.
Chapter 7. Credential Sharing on Phishing Sites

to phishing defenses by searching for phishing kits, and take action to block such sites, or issue a warning to the site moderator (in case they were compromised).

**Using third-party Sharing Channel for Defense.** We believe that third-party sharing (and backdoors) can also be used by defenders for good purposes. For example, for known third-party collectors (backdoor email addresses or client-side collectors), instead of directly shutting them down, the defenders (e.g., law enforcement, service providers) may keep them alive but take away the ownership from the malicious parties. For example, Google can block the attacker from accessing the Gmail account that acts as the backdoor collector. Then Gmail’s security team can keep this account alive as a vantage point to monitor the phishing activities from the same class of phishing kits. The benefit is that whenever the corresponding phishing kits are used to perform phishing in the wild, the defenders can directly pinpoint the location of the attackers (since the phishing kits will contact the backdoor collector). In addition, the defender will also receive a copy of the victim list, which allows defenders to take early actions to alert the victims.

**Limitations.** Our study has a few limitations. *First*, while we obtain a complete view of client-side sharing, we still do not have the complete picture on the server-side. We only observe instantaneous sharing of credentials on the server-side, i.e., as soon as the credentials are received by the server. This is a limitation because it is still possible that the server-side scripts may send credentials at a later point of time, e.g., based on pre-set timers. Unfortunately, given the large number of phishing kits we need to test, we cannot monitor them for a long time. *Second*, our server-side analysis is based on the phishing kits—we have no information about phishing sites that do not leave kits publicly accessible. *Third*, we acknowledge that our dataset is biased due to the use of the four phishing blacklists which are skewed towards English speaking countries. However, our dataset still covers phishing sites that target major sectors and a broad set of brands (Table 7.5). *Fourth*, our view of post-phishing activities is limited due to the small scale of the experiment. For ethical concerns, the small scale is intended.

### 7.9 Related Work

**Password Leakage.** While existing works have studied password leakage [89] and password re-use [106, 267, 291], credentials sharing during the phishing process wasn’t well understood. A related study [274] examined the potential victims of off-the-shelf keyloggers, phishing kits and previous data breaches. They explored how stolen passwords enabled attackers to hijack Gmail accounts.

**Phishing Kit.** Zawoad et al. found 10% of phishing sites had evidence of using phishing kits [314]. Phishers’ motivation and thought processes are inferred by analyzing phishing kits [52, 100, 193, 228]. Previous work has also sandboxed phishing kits to monitor their
mechanisms and behavior of criminals [143]. Phishers usually use phishing kits to create a series of similar phishing pages [85].

**Phishing Detection & Warning.** Content-based detection methods have been studied extensively. Previous researches have studied anomaly detection as a service [305, 306, 307, 311, 318]. Cantina and Cantina+ [304, 322] base their detection on DOM and search engines information. Researchers also looked into other detection methods based on visual similarities [297], URL properties [79, 205, 277], OCR features [61, 113], and user behavior patterns [111, 264]. Going deeper, phishing hosts have also been extensively studied including compromised sites [107] and malicious web infrastructure [197]. Phishing emails are used to distribute phishing URLs. Phishers can use email spoofing techniques [152, 155] or email header injection [244] to deceive users. Other researchers looked into the effectiveness of phishing websites warning and prevention in web browsers [65, 117, 300]. A key novelty of our work is to track the information flow for credential sharing across different phases of phishing.

### 7.10 Conclusion

In this chapter, we performs an empirical measurement on the information flows of credential sharing during phishing attacks. Our analysis covers more than 179,000 phishing URLs (47,000 live phishing sites). We show that user credentials are shared in real-time to multiple parties at both the client side and the server side. Although third-party sharing exposes user credentials to even more malicious parties, we argue that defenders may make use of these channels to back-track phishing servers and alert phishing victims.
Chapter 8

Sensitive Applications of Voice Personal Assistant Systems

8.1 Acknowledgement

This project is a collaboration with Dr. Yuan Tian and Faysal Hossain Shezan from University of Virginia. The leading author of this work is Faysal Hossain Shezan. I am the second author of this work. Faysal Hossain Shezan came up the original idea of this project and led the execution of experiments and most data analysis and paper writing. I helped by collecting skill dataset by crawling Alexa and Google Assistant skill stores. I also developed a tool for the sanity check which can be used to interact with the Alexa cloud service using text input and output.

8.2 Introduction

The ubiquitous usage of the Internet of Things (IoT) devices has proliferated the number of Voice Personal Assistant (VPA) systems in our home. As of Jan 2019, over 66.5 million households in the US [3] have one or more VPAs such as Amazon Alexa [67], Google Home [22], and Homepod [25]. The two dominating manufactures Amazon and Google introduce the voice assistant applications called “skills”. Third-party developers have built and published more than 84,000 skills worldwide in the application markets in 2019 [9, 20]. Users can “talk” to these applications to complete various tasks including opening a smart lock, starting their car, placing shopping orders, and transferring money to a friend. Although these applications bring convenience, they also introduce new attack surfaces. Recent research shows that remote attackers can craft hidden voice commands to trigger the VPAs to launch malicious actions without user knowledge [254, 312, 316]. More recent work shows that attackers can publish malicious skills with similar pronunciations to fool the VPA to invoke the wrong application [182, 320]. Existing works have focused on proof-of-concept attacks by pointing out the potential ways of launching the attacks. However, there is a lack of empirical understanding of what functionality the third-party applications provide, and thus makes it difficult to systematically assess the consequences of these attacks.

1Google calls the applications as “actions”. For consistency, we also call them as skills.
In this chapter, we perform the first large-scale measurement on the third-party applications of Amazon Alexa and Google Home to systematically assess the attack surfaces. More specifically, given a voice assistant application, we seek to characterize its risk by detecting and analyzing the sensitive voice commands that are subject to potential attacks. Based on the recent proof-of-concept attacks [182, 254, 312, 316, 320], there are two main types of attack consequences: (1) controlling the system to perform an action, and (2) obtaining sensitive information. As such, we develop a natural language processing tool that classifies a given voice command from two dimensions. First, we examine whether a voice command is designed to insert an action (e.g., controlling a smart device) or retrieve information (e.g., obtaining user bank balance). Second, we classify whether the command is sensitive or nonsensitive. These two dimensions help to provide a more comprehensive view of the voice assistant skills, and their susceptibility to the existing attacks.

**Challenges.** There are four key challenges to automatically analyze the functionality of VPA skills. *First*, unlike smartphone apps whose binaries (or source code) are available for analysis, voice applications are essentially web programs that are hidden behind the cloud (e.g., Amazon/Google cloud). Thus, we cannot characterize the voice skills using traditional API analysis but need to design new tools to analyze its natural language interface (or voice commands). *Second*, the voice commands supported by VPA skills are very short, which provides little information to run typical Natural Language Processing tools. *Third*, there are already a large number of VPA skills in the current markets, and labeling their data (for model training) requires expensive manual efforts. *Fourth*, the perceived sensitivity of a voice command could vary from person to person, the measurement of which requires user participation.

**System Design.** To automatically analyze the voice commands, we design two classification models to characterize the *capability* and the *sensitivity* of the voice commands respectively.

First, regarding the voice command’s capability, we train a model to classify *action injection* commands that control smart devices and services, *information retrieval* commands that retrieve information from the application. For example, “Alexa, ask Watch-Man to open the red door” is an injection command, while “Alexa, ask Macys where is the nearest store to me” is a retrieval command. Our model is based on a Convolutional Neural Network (CNN) [170]. To overcome the short length of each command, we append the skill category information to provide contexts. In addition, we design an active learning-based workflow so that we can minimize manual labeling efforts to train an accurate model. The ground-truth evaluation shows that our model achieves an accuracy of 95%.

Second, regarding the voice command’s sensitivity, we build a model to classify *sensitive* commands from *nonsensitive* ones. For example, “Alexa, unlock my front door” is a sensitive command while “Alexa, play Harry Potter Quiz” is nonsensitive. The challenge is that sensitivity classification is rather subjective, and conventional user studies have limited scalability.
As such, we use automated algorithms for sensitive keyword extraction and then perform a
user study (N=404) for keyword pruning. Instead of using complex machine learning models
(whose results are difficult to interpret during post-analysis), we use a keyword-based model
that achieves an accuracy of 95.6% in finding the sensitive voice commands.

**Measurement Results.** We apply this tool to analyze 77,957 Amazon Alexa skills and 4,813 Google Home skills over two years (2018-2019). We identify 19,263 sensitive “action injection” commands and 5,352 sensitive “information retrieval” commands. We find these sensitive voice commands are from a small set of 4,596 skills (4,203 Alexa skills and 393 Google Home skills), which only take 5.55% of all the available skills. Note that there are some duplicated skills and voice commands for 2018 and 2019. After removing the duplicates (6,058 sensitive commands and 1,216 sensitive skills) between Alexa 2018 & Alexa 2019, and duplicates (40 sensitive commands and 165 sensitive skills) between Google 2018 & Google 2019, we identify 18,517 unique sensitive voice commands (16,844 from Amazon Alexa and 1,673 from Google), and 3,215 unique sensitive skills (2,987 from Amazon Alexa and 228 from Google). 90.46% of these sensitive commands are from skills that are used to communicate with smart-home devices. Surprisingly, categories that are traditionally perceived to be sensitive (e.g., “Health” and “Kid”) rarely have sensitive voice commands and skills. Even the “Shopping” category only contributed 146 sensitive commands across the two platforms. We also find that the sensitive voice commands are highly concentrated on a few sets of “actions”. The top 30 sensitive keywords effectively cover 98.7% and 99.3% sensitive commands in Amazon Alexa and Google Home respectively. Overall, the results show that despite a large number of available skills (over 82,770), only a small portion of skills are for security and privacy-sensitive tasks that deserve further attention from researchers for security analysis. However, the number of sensitive skills and voice commands increase from 2018 to 2019 (907 new sensitive skills, and 6,088 new sensitive commands).

**Summary of Contributions.** Our key contributions are:

- **First**, we perform the first large-scale empirical measurement on two dominating Voice Personal Assistant application markets, covering 82,770 skills and 211,843 voice commands.

- **Second**, our results provide new understandings of the capability and the sensitivity of the third-party applications. We identify a small set of sensitive applications (5.55%) that contributed to the vast majority of sensitive voice commands.

- **Third**, we design and implement automated tools to classify VPA skills and their voice commands [28].

- **Fourth**, we perform a user survey with 400+ participants to measure the perceived sensitivity of voice commands.
8.3 Problem Definition and Goals

In this section, we present the threat model and our research questions regarding the attack surfaces of VPA systems.

8.3.1 Threat Model

Researchers show different proof-of-concept attacks that can exploit the VPA ecosystem. Recent papers demonstrate that remote attackers can send malicious voice commands to control the devices stealthily [312]. Attackers can also create malicious skills whose names have similar pronunciations with those of popular skills, as a way to trick users to invoke malicious skills without their knowledge [182, 320]. Recent incidents [11, 14] also show that background noise or TV commercials can trigger unwanted actions in the VPA system. Despite the proof-of-concept, it is not yet clear what consequences these attacks can cause. The reason is that we still lack the understanding of what existing skills are capable of doing, and how sensitive their tasks are. As such, in this chapter, we seek to measure the capability and sensitivity of the voice commands and the skills. In our threat model, the VPAs are trusted. We focus on two types of attacks from external attackers:

Hidden Voice Command Attack. Remote attackers can send malicious voice commands to trigger the VPAs for malicious actions. The voice commands can be sent through compromised local speakers or embedded in TV commercials and popular music. By utilizing the feature that humans cannot hear the high-frequency sound, an attacker can trigger the malicious voice commands without the user’s notice [254, 316]. As illustrated in Figure 8.1a using inaudible voice command, an attacker can trigger malicious events in smart home devices such as unlock the smart door lock. Moreover, an attacker can record sensitive information (such as account balance, PIN code) if they also compromised microphones [31] in the home.

(b) Skill Squatting Attack: Attacker registering a malicious skill whose name sounds like that of the popular “PayPal” skill. The cloud could misinterpret the user voice command to invoke the malicious skill.

Figure 8.1: Two types of attack against the VPA system.
Skill Squatting Attack. Attackers can develop and publish a malicious skill to collect sensitive user information. The key idea of skill squatting is to register the skill with a name that sounds similar to the target skill. As is shown in Figure 8.1b, the Cloud may misinterpret the voice commands and invoke the malicious skills instead of the legitimate skills since their names sound similar. Then attackers can collect sensitive information (such as PIN code, PII) as users think they are interacting with the legitimate PayPal skill.

8.3.2 Analysis Goals

Given a skill and its voice commands, we seek to understand the capability and sensitivity of each voice command, to better understand the potential consequences caused by the above attacks. Note that this analysis is different from analyzing the data collection controlled by permissions. Alexa and Google Home also have permissions similar to smartphones, but these permissions are limited, and only protect information from Amazon or Google account (e.g., user’s zipcode). We focus on a different and more common way of data collection, where skills get the information from users directly via voice interfaces, instead of via Amazon or Google’s account.

Capability Measurement. We classify the voice command’s capability based on whether it is used to inject actions, or retrieve information. On one hand, an action injection command can be directly exploited by hidden voice attacks to insert an action (e.g., unlocking doors, placing shopping orders) without user knowledge. On the other hand, an information retrieval voice command can be used for collecting sensitive information or return users’ wrong information (e.g., fake news). For commands that get information from the skill, the skill squatting attacker can pretend to be a benign skill and give fake information. Since users are interacting with the malicious skill (without knowing the skill is the wrong one), they would trust the fake information from the malicious skill. Besides, an attacker can launch the hidden voice command attack to invoke a benign skill and send an information retrieval command secretly to the VPA, for example, “what is my PIN code?”, When the VPA reply with sensitive information, the attacker can record the information by a compromised speaker.

Sensitivity Measurement. We seek to investigate whether the voice command is sensitive. Regardless of the capability (action injection or information retrieval), certain voice commands do not carry real risks, especially for simple skills without account linking, e.g., games and quizzes. Our goal is to differentiate the sensitive commands (e.g., unlock the front door) with the nonsensitive ones (e.g., tell me some jokes) by considering user perceptions. In this work, we identify those voice commands as sensitive if exploited they will bring damage to the user by either leaking private information (e.g., bank balance, inbox message) or violating security (e.g., unlock the front door, stop recording camera) via hacking IoT devices. In contrast, according to our definition, nonsensitive voice commands do not pose security or privacy threat if exploited. Voice commands that give general information...
<table>
<thead>
<tr>
<th>Skill Name</th>
<th>Command Line</th>
<th>Type</th>
</tr>
</thead>
</table>
| Blink SmartHome | “Stop the camera”  
                            “Show me the last activity from front door” | Injection  
                            Retrieval |
| Schlage Sense | “Lock the door”  
                            “Ask FordPass to start my car”  
                            “Tell FordPass to list all cars on the account” | Injection  
                            Retrieval |

Table 8.1: Example skills and their sensitive voice commands.

(e.g., restaurant information, weather information) or use to operate third party applications that have no security implication generally fall into nonsensitive class (e.g., tell a joke, play rock music).

### 8.3.3 Motivating Examples

Table 8.1 lists three example skills and their sensitive commands. **Blink** is a home security skill that controls the cameras and alarms. For example, the user can change the home security mode by saying an injection command “Alexa, ask Blink to arm/disarm my home system”, and check the camera feeds by a retrieval command “Alexa, show me the last activity from the front door”. The underlying danger is that the security mode can be changed by attackers and it might release the user’s recorded video information. **Schlage Sense** controls the smart locks on the doors. The user can use this skill to lock/unlock the door by the injection command “Alexa, lock/unlock the front door” and check the door status by the retrieval command “Alexa, is the front door locked?”. The possible threat is that this skill gives incorrect door status to the user, leaving the user’s home in a dangerous situation. **FordPass** is a skill to control network-connected cars. Users can control the car by injection commands “Alexa, ask FordPass to start/stop my car”, and obtain vehicle information by the retrieval command “Alexa, ask FordPass my tire pressure”. Our goal is to identify and characterize these sensitive voice commands that are likely subject to attacks. More examples of sensitive and nonsensitive voice commands are shared via the following link [19].

### 8.4 Experiment Design

In the following section, we will describe the details of our data collection, system design, and experiment methodology.
Chapter 8. Sensitive Applications of Voice Personal Assistant Systems

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Skill</td>
<td>31,413</td>
<td>20,213</td>
<td>26,331</td>
<td>3,148</td>
<td>1,665</td>
</tr>
<tr>
<td>Command</td>
<td>80,129</td>
<td>51,922</td>
<td>66,148</td>
<td>9,096</td>
<td>4,548</td>
</tr>
</tbody>
</table>

Table 8.2: Our dataset.

8.4.1 Data Collection

For our analysis, we collected data for existing applications in the Alexa store [6] and Google Home store [23]. The goal is to cover a large number of applications with a priority for the popular ones. First, given an app store, we started from the homepage to identify all the application categories (23 categories in the Alexa store, and 18 categories in the Google Home store). Then, we crawled all the indexed skills and their introduction page under each category. Each skill’s introduction page contains the skill name, skill description, category information, developer information, user rating and reviews, privacy policies, and the supported voice commands.

We crawled five datasets during 2018 and 2019 (Table 8.2). More specifically, we crawled the Alexa US store and Google Home store in June 2018, and later again in May 2019. Since Alexa has region-based skill markets, to compare the difference of skills in different regions, in August 2019, we also crawled a snapshot of Alexa UK (United Kingdom) store for comparison purposes (Google Home only has one universal app store). Note that even for the same store, the different snapshots do not necessarily contain the same set of skills. For example, comparing Alexa US 2019 and Alexa US 2018, there were only 18,309 (58.28%) skills in both snapshots as new skills are entering the stores while old skills disappearing. Even for skills in both snapshots, we found developers may update the skill description and example voice commands. In total, there are 38,093 (47.5%) overlapping voice commands in 2018 and 2019 Alexa US data. Moreover, we observed 1,060 skills and 1,441 voice commands from Google 2018 also appeared in Google 2019 data.

In total, there were 82,770 skills (77,957 from Amazon Alexa and 4,813 from Google Home) in the five datasets. By referencing the publicly reported statistics on Amazon Alexa and Google Home stores [8, 258], we believe our datasets are rather complete. In the following, for the ease of presentation, we will mainly use the datasets in the US store 2019 to present our findings. To show the evolution of the skills from 2018-2019 and the skills in different regions, we will present these results in the Section 8.5.3 evolution analysis, and Section 8.5.3 region-based analysis.

Extracting Voice Commands. As previously mentioned, each skill page displays several example voice commands, which are easy to extract. However, Amazon Alexa only allows showing up to three voice commands on the page [6] and Google Home allows showing up to five voice commands [21]. Due to this limit, we find that developers often include
additional voice commands in the skill description as a *list* or in a *double quote*. To extract voice commands from the skill description, we follow the steps below.

1. We convert the text to the lowercase format, convert Unicode objects to ASCII strings, and remove special characters.

2. We divide the description into different chunks, based on a line break, newline, and double-quote ("").

3. If the text chunk starts with the *wake word* (i.e., “alexa,” “google,”), then we mark that text chunk as a voice command.

Table 8.2 shows the number of voice commands extracted from each dataset (combining example commands and commands in the description). In total, we extracted 211,843 voice commands. While we cannot guarantee the voice command extraction is exhaustive, we argue that developers are motivated to put the most essential commands on the skill page to showcase the skill’s functionality and teach users how to use the skill. Later in Section 8.4.6, we will run a dynamic interaction experiment with skills, and show that our voice command extraction is already rather complete.

### 8.4.2 Data Pre-processing

Before building the classification models, we first pre-process the voice command datasets to produce a clear format. We use the Amazon Alexa US 2019 and Google Home 2019 as the primary datasets to explain the following steps.

1. **Removing voice commands used for enabling skills.** These voice commands are used to turn on (or invoke) the skill on the user device (e.g., open *Amex*, start *Song Quiz*). We remove them since they don’t indicate the function of the skill or perform any actions. As a result, we remove 29,784 voice commands for Amazon Alexa, and 2,949 voice commands for Google Home.

2. **Extracting action from a voice command.** The action of a voice command refers to a user request. According to the developer guide [15, 16], voice commands must follow the general patterns (defined by Amazon and Google) as follows:

   - `<action> <connecting word> <invocation name>`
   - `Ask <invocation name> <connecting word> <action>`
   - `Ask <invocation name> < action>`
• Ask <invocation name> <question beginning with a supported question word such as ‘what’, ‘how’, etc.>

Based on the above rules, we extract actions from voice commands. For example, for command “Ask Mastermind to text Kelly Miller”, we first tag “Mastermind” as the invocation name, and then identify the connecting word “to”. After that, we find the action word, which is “text Kelly Miller”.

3. Data structure formatting. In this step, we remove punctuation, convert all the characters to the lowercase, and convert numeric value to the corresponding alphabetic value (e.g. 1 to “one”, 2 to “two”). We also replace all the invocation name with a general name. For example, “ask mastermind to text Kelly Miller” will be converted to “ask invk_name to text Kelly Miller”. This step is to remove potential distractions for the classification models.

4. Adding category name. Voice commands are usually too short and lack the necessary context. To mitigate it, we concatenate the skill category to the voice command to provide the context.

5. Removing redundancy. For example, “CNN Flash Briefing”, “ABC News Update” and “Fox News” are three news skills who have the voice command “what’s in the news?”. We remove such identical voice commands to avoid biases of the trained models. Note that after the pre-processing steps above, certain previously non-identical commands become identical now. For example, “ask Doctor Who Facts for a fact” and “ask Unofficial Stargate Facts for a fact” become the same command after replacing invocation name with a common term (invk_name) in step-3. In total, we remove 1,141 duplicate voice commands for Amazon Alexa and 296 duplicate voice commands for Google Home.

8.4.3 Data Labeling

Our model training and evaluation require “ground-truth” data, and we create a small ground-truth dataset by manually labeling voice commands. Our model training is primarily based on the Alexa US 2019 data. Then, we evaluate the trained model on both Alexa US 2019 and Google Home 2019 data.

For training the capability analysis model, we randomly select 862 Amazon Alexa skills covering all 23 categories, and label the 1,810 voice commands as “action injection” or “information retrieval”. To validate model performance, from Amazon Alexa we randomly select another 247 skills and label 475 voice commands. From Google Home, we randomly select 87 skills and label 200 commands.

Similarly, for training the sensitivity analysis model, we randomly select 721 skills from Alexa and label 1,652 voice commands into “sensitive” and “nonsensitive”. For model validation, we select another 99 skills from Alexa and label 275 voice commands. From Google Home,
we randomly select 83 skills and label 200 commands. We have three researchers to label the voice commands. Each researcher works independently on the labeling tasks. If the three researchers label a command differently, we use the majority voting to resolve the conflict. For the labels on “action injection” and “information retrieval”, we have very consistent labels across voice commands (agreement rate = 97.33%, Fleiss’ kappa = 0.903) [129]. For the labels regarding sensitivity, the researchers have slightly bigger differences (agreement rate = 94.46%, Fleiss’ kappa = 0.93). Because the sensitivity label is rather subjective, we conduct a user study to further validate and calibrate the sensitivity assessment (details are explained in Section 8.4.5).

8.4.4 Experiment 1. Capability Analysis Model

To classify if a voice command is action injection or information retrieval, we apply active learning [132, 211, 276, 310] to achieve our targeted accuracy with limited labeled data. Transfer learning might be another option for dealing with limited labeled data [151, 239, 245, 313], but in our case, active learning turns out to be a better solution because it is challenging to find a relevant source domain.

We use Convolutional Neural Network (CNN) for building the core machine learning model in our active learning approach. Applying CNN to natural language processing, especially text classification has been proven effective compared to the other DNN (Deep Neural Network) models [170, 195]. We build our embedding layer from word2vec [236] model using 80,129 unlabeled voice commands from Alexa US 2019 data. Our model classifies the outcome of the inputted voice command using a softmax dense layer, predicting its capability category.

Algorithm 1 shows the procedure of model training. Let us first illustrate some preliminaries and notations. \( \mathcal{L} = \{x_i, y_i\}_{i=1}^m \) denotes the training dataset which contains \( m \) number of labeled instances, whereas \( \mathcal{U} = \{x_i\}_{i=m+1}^n \) indicates the set of unlabeled instances. Here, \( x_i \in \mathbb{R}^d \) is a \( d \)-dimensional feature vector and the class label, \( y_i \in C = \{0, 1\} \), where zero and one represents action injection, and information retrieval class respectively. We consider achieving a targeted accuracy as a stopping criterion for active learning [192, 323]. For each round \( R \), we calculate the uncertainty value of instances from the unlabeled dataset according to Equation 8.1 in Line 8-10 of Algorithm 1.

\[
P_L(y_i|x_i) = |p(0|x_i) - p(1|x_i)| \tag{8.1}
\]

In Equation 8.1, \( P_L \) indicates the uncertainty value of an instance, and \( p \) denotes the probability of an instance being in one class.

\[
\{x^*_R\} = \text{argmin}_{100}(P_L(y_i|x_i)) \tag{8.2}
\]

We then select those instances for which the model is mostly uncertain and label these
Algorithm 1 Retraining-based Active Learning algorithm for Capability Analysis Model

1: **Input:** Labeled data set \( L \), unlabeled data set \( U \), validation set, \( V \)

2: **Output:** Trained Model, \( M_R \) and Model’s Accuracy, \( A \)

3: **procedure** TrainCapabilityAnalysisModel

4: Train the classifier on \( L \)

5: \( R \leftarrow \) initialize to zero

6: repeat

7: \( R \leftarrow R + 1 \)

8: for each instance, \( x_i \in U, y_i \in C = \{0, 1\} \) do

9: calculate uncertainty, \( P_L(y_i| x_i) \) using Equation 8.1

10: end for

11: Choose uncertainty set, \( \{x^*_j\}_{j=1}^{100} \) using Equation 8.2

12: Construct newly labeled set, \( \{x^*_j, y^*_j\}_{j=1}^{100} \)

13: \( L \leftarrow L \cup \{x^*_j, y^*_j\}_{j=1}^{100} \)

14: \( U \leftarrow U \setminus \{x^*_j\}_{j=1}^{100} \)

15: Re-train the model using \( L \)

16: Compute accuracy, \( A \) of the current model, \( M_R \) on \( V \)

17: until \( A > 95\% \)

18: Return Trained Model, \( M_R \) and Model’s Accuracy, \( A \)

19: **end procedure**

instances manually. At each round, we select 100 most uncertain data, \( \{x^*_j\}_R \) according to Equation 8.2. Then, we remove \( \{x^*_j\}_R \) from \( U \) and label \( \{x^*_j\}_R \) by human to get \( \{x^*_j, y^*_j\}_R \). And thus, we get additional labeled data \( L \leftarrow L \cup \{x^*_j, y^*_j\}_R \) to train the model in the next round. We keep iterating this process until we reach our targeted accuracy. We compute our model’s performance on a fixed validation data, \( V = \{x_j, y_j\}_{j=1}^{475} \). We stop this data augmentation process as soon as our model reach our expected performance. Since active learning is an iterative process, we need to decide when to stop the process. The most popular techniques for active learning stopping criteria are - achieving targeted accuracy, number of total iterations, gradient-based stopping, confidence-based stopping, performance degrading point, achieving stable performance [192, 323]. We set the targeted accuracy as 95% to be the stopping criteria.

8.4.5 Experiment 2. Sensitivity Analysis Model

We observe that it is intuitive to estimate the sensitivity of voice commands of account linking required skills based on the presence of certain keywords. Because such skills usually have more sensitive functionality (e.g., unlock the door, stop recording camera, etc.) compared to the non-account linking skills. With this observation, we design the keyword-based solution to identify sensitive voice commands. Figure 8.2 illustrates our complete keyword-based search technique. We can divide our whole keyword-based search technique into three
8.4. Experiment Design

Figure 8.2: System Overview for the keyword-based approach of finding sensitive and nonsensitive voice commands.

different phases- (1) Sensitive keyword extraction, (2) Expanding sensitive keyword set, and (3) Fine-tune keyword set using user survey. The first phase is used for extracting sensitive keyword set from voice commands while the second phase is used for increasing that sensitive keyword set. As sensitiveness is subjective, our third phrase includes an online survey to collect feedback on the keywords that are selected by our tool. Note that, the online survey is approved by IRB (Institutional Review Board). Finally, we get the ultimate sensitive keyword set that we use for evaluating the sensitiveness of a voice command.

Sensitive Keyword Extraction. Initially, we have three sensitive keyword sets $A_1, A_2, A_3$ (identified by three annotators while labeling sensitive voice commands). After aggregating all the sensitive keywords, we have the initial keyword set, $K_I = A_1 \cap A_2 \cap A_3$. Then, we use RAKE (Rapid Automatic Keyword Extraction) [253] for extracting sensitive keywords, $R_S$ from sensitive voice commands, $S$ and nonsensitive keywords, $R_N$ from nonsensitive voice commands, $N$. We remove those sensitive keywords from $R_S$ that are also present in $R_N$. And thus, we compute keywords, $R'(= R_S - R_N)$ unique to sensitive voice commands. Finally, we get the initial sensitive keyword set (which contains 68 sensitive keywords).

Sensitive keyword set expansion. To decrease the false-negative ratio, we expand our sensitive keyword set by using word2vec [236]. To train the word2vec model, we use 80,129 unlabeled voice commands from Alexa US 2019 data to get expanded keyword set, $R'E$. These keywords are semantically similar to our sensitive keywords, listed in $R'$. In this way, we get 38 new sensitive keywords. As a result, the size of our total keywords, $R'' (= R' \cup R'E)$ become 106.

Fine-tune keyword set using survey. We launch an online survey to get a fine-tuned keyword set. From the survey, we get our final keyword set that contains 57 sensitive keywords.

Survey Design. We launch our survey through Mechanical Turk [7] to collect data from a larger population. We collect three different categories of information: how often users use VPA, users’ opinions on sensitive keywords, and users’ demographic information.

From the first two steps of the keyword-based approach, we get 106 sensitive keywords. Listing this large set of keywords in the survey might introduce boredom to the participants. That’s why we divide the keyword list into two subsets before running the survey. First,
Chapter 8. Sensitive Applications of Voice Personal Assistant Systems

to find out whether user uses voice assistant or not, we begin with simple questions; e.g. do users’ ever use any voice personal assistant (e.g., Google Home, Amazon Alexa) or, do they use voice assistants (e.g, SIRI, Cortana, Google Assistant) in their mobile devices, and also how long they have been using it. Then, we show them a list of voice commands highlighting the functional keywords which represent the functionality of the voice commands. For example, for the following command—‘Alexa, tell Virtual Keypad to “arm” my system away’, we highlight functional keyword “arm”. Next, we ask users to classify functional keywords into five different scales of sensitiveness, including - not sensitive, less sensitive, neutral, sensitive, and most sensitive. We also include the “other” option for each of the voice commands where the user can list different functional keywords if she feels the highlighted one does not represent the functionality of the voice command. Finally, we collect users’ demographic information, such as age, gender, education level, and occupation. We have uploaded all of our survey questions via these links [39, 40].

\[
\varphi(x) = \begin{cases} 
1 & \text{if } (#\text{most sens.} + #\text{sens.}) > (#\text{not sens.} + #\text{less sens.} + #\text{neutral}), \\
0 & \text{otherwise}
\end{cases}
\]  

(8.3)

Survey Results and Analysis. In our survey, we also included an attention check question to identify invalid responses. After removing 221 invalid responses, we collected a total of 404 valid responses (199 for part-I and 205 for part-II). In the following, we focus on understanding users’ responses to our sensitive keyword list. Participants can classify a keyword into five different scales (not sensitive, less sensitive and neutral are counted as a nonsensitive vote, whereas sensitive and most sensitive are counted as a sensitive vote). According to the Equation 8.3, we decide whether a keyword belongs to a sensitive class or nonsensitive class. If \(\varphi(x)\) is equal to one for a particular keyword, then it is considered as a sensitive keyword, otherwise, that is categorized as a nonsensitive keyword. After this process, the size of our final keyword list becomes 57. Our survey results show that most participants are likely to agree with the sensitive keywords we generate in the first two stages (sensitive votes and most sensitive votes add up to 53.8%). We have uploaded all the sensitive keywords to this link [38].

8.4.6 Sanity Check for Undocumented Voice Commands

In this section, we conduct an extra step to uncover “undocumented” voice commands supported by skills. By “undocumented” we refer to those voice commands that are not listed in skill’s recommended voice command list and description. We suspect that malicious developers may hide sensitive voice commands by not revealing to the user. In the future, by triggering those commands through a hidden voice command attack (Figure 8.1a), the attacker can steal user information or execute sensitive operations.
8.5. Measurements and Evaluations

Tool for interacting with skill

Currently, we only focus on uncovering the undocumented voice commands for Amazon Alexa (US 2019) because it has 24 million more users than Google Home [3].

Challenges. An intuitive method is to use a speaker to play commands to an Alexa device and use a microphone to record audio responses from the Alexa device. However, this is time-consuming and requires a quiet room for better recognition of commands.

Virtual Alexa Client. Therefore, we introduce our lightweight testing tool that does not require any physical setup. The testing tool has two modules: a virtual Alexa client (which works the same as physical Alexa device) and a speech-to-text module. Alexa provides a testing interface [10] to help developers to test their skills and other skills on the market with text input. Our virtual Alexa client utilizes this feature to test skills with text input instead of voice input. After receiving the input, the virtual client returns a series of responses either in text format or in audio format. We use the Azure speech-to-text service [13] to transcribe audio responses.

Seed sensitive command for testing

Next, with the automated interaction tool, we test if skills support sensitive commands.

Categories and Commands. For doing the sanity check, we use our keyword-based model to identify sensitive voice commands. Unfortunately, 2,782 number of skills require account linking, and 1,657 of them are from the smart home category. As a result, we are unable to do the sanity check in this critical category due to the lacking of legitimate information. We select top 154 sensitive voice commands (based on the number of occurrences) from the following categories—shopping, communication, productivity, business & finance, and health & fitness. Because undocumented voice commands from these categories can damage a lot compared to others. We randomly choose 50 skills from those five categories, and investigate potential undocumented voice commands (250 skills in total). We run our experiment by taking each of the voice commands from each category, and test it with all the skills in that category.

8.5 Measurements and Evaluations

We evaluate the performance of our method and find it effective for both the capability analysis (F1-score is 96.33%) and sensitivity analysis (F1-score is 91.78). We also run a systematic measurement of voice commands, including perspectives such as cross-platform analysis, category analysis, evolution analysis, and region-based analysis. With all the data we analyzed, we have found 5.55% (4,596 out of 82,770) skills are sensitive, and 11.62%
Table 8.3: Ground-truth data of Amazon Alexa US 2019 for Capability and Sensitivity analysis model.

<table>
<thead>
<tr>
<th>Capability</th>
<th>Action Injection</th>
<th>Information Retrieval</th>
<th>2,285</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensitivity</td>
<td>Sensitive</td>
<td>515 (26.72%)</td>
<td>1,927</td>
</tr>
<tr>
<td></td>
<td>Nonsensitive</td>
<td>1,412 (73.27%)</td>
<td></td>
</tr>
</tbody>
</table>

(24,615 out of 211,843) voice commands are sensitive. We show the details of our evaluation and measurement results below. For ease of presentation, we report the results on US store data in 2019, and compare the results with 2018 data, and UK data in Section 8.5.3 and Section 8.5.3 respectively.

8.5.1 Capability Analysis Model

We achieved an accuracy of 95% for the capability analysis of commands with the active learning approach described in Section 8.4.4.

As described in Section 8.4.3, three annotators label 2,285 voice commands as action injection and information retrieval in Amazon Alexa. A complete overview of the labeled data for the capability analysis model is shown in Table 8.3. From the labeled data, we randomly select 247 skills (which include 475 voice commands) as the validation dataset, and use the rest of the 862 skills (1,810 voice commands, 501 as action injection, and 1,309 as information retrieval) to train the model. We use the validation set for each round of the active learning process to evaluate our model’s performance. Note that the validation set never overlapped with any of the training data. We reach our optimal point at round ten. And thus, we stop our active learning procedure at that point. Finally, we get our improved model $\mathcal{M}$ that has an accuracy of 95.16% (94.68% precision, 98.04% recall, and 96.33% F1-score) over the validation data.

To investigate if our model is biased towards this validation set, we extend our experiments to run more evaluations. We randomly pick 300 voice commands from the unlabeled dataset. Then, we make predictions using our model, $\mathcal{M}$ on those data. Finally, we verify those predictions by human, and find that our model’s accuracy was 94.65% which is close to the validation accuracy.

We also evaluate our model’s performance on Google data. To validate the performance, we label 200 voice commands from 87 voice applications (49 are action injection, 151 are information retrieval) from Google Home. Our active learning model achieves 95.99% Accuracy (whereas, 99.32% Precision, 95.38% Recall, and 97.31% F1-score) while running capability analysis on Google data. We compare our model’s performance with four baselines. In Figure 8.3, we present these baseline models’ performances along with our active learning using the margin sampling technique. The fifth bar is our proposed scheme, which outperforms all
Figure 8.3: We compare our active learning using margin sampling model’s performance with four different baseline approaches including-(1) Base RNN; where we use RNN network structure for building the machine learning model, (2) Base CNN; where we use CNN network structure, (3) CNN + data clean; where before training the model, we process the input data according to Section 8.4.2, (4) CNN + ActiveL (Entropy); where we use entropy metric to select unlabeled data to be labeled in each round of active learning approach, (5) CNN + ActiveL (Margin) is our proposed method; where we select the most uncertain unlabeled data to be labeled in each round of active learning approach (sorted based on F1-score).

8.5.2 Sensitivity Analysis Model

Based on our keyword-based strategy (described in Section 8.4.5), we can classify the sensitivity of commands with an accuracy of 95.6%.

For sensitivity analysis, we label 1,927 data (247 Alexa skills) from different categories as illustrated in Table 8.3. Among them, we randomly select 1,652 data (sensitive data: 508, nonsensitive data: 1,144) for building the sensitive keyword set. And the rest of the data (e.g., 275 voice commands from 99 skill) are chosen for evaluating the model’s performance. This validation set has never been considered while building the keyword list.

The performance of our keyword-based approach depends on correctly identifying the sensitive keywords. One major part of our keyword-based approach is expanding the sensitive
Chapter 8. Sensitive Applications of Voice Personal Assistant Systems

keyword set using word2vec. While finding similar sensitive keywords, we set the cosine similarity value to 0.8 because of two reasons. First, by varying cosine similarity value from 0.5 to 0.95, we find that 0.8 works the best. Second, we perform a case study on the keyword pairs with different cosine similarity. For 0.8, we find several pairs such as (arm, disarm), (arm, activate), (increase, decrease), (dim, brighten). In a single tuple, the first word is the source word while the second one is a similar word found by word2vec. We can observe that the similar keywords also represent the sensitive characteristics (according to Section 8.3.2), e.g.- disarm the alarm, activate the climate control. However, if we lower the threshold and set it to 0.75, it would give us similar keyword pairs, such as- (decrease, seventy-five), (dim, percent), which no longer represent the sensitive characteristics. Finally, using this keyword-based approach, we get an accuracy of 95.6%, precision of 95.71%, recall of 88.16% and F1-score of 91.78% over the Alexa validation dataset.

To evaluate our model’s performance on Google data, we label 200 voice commands from 83 applications (57 are sensitive, 143 are nonsensitive). Our model achieves 96.5% Accuracy (whereas, 93.44% Precision, 95% Recall, and 94.21% F1-score) on Google data.

8.5.3 Measuring Security Implications of Skills.

Now, we put together the results from the capability and sensitivity analysis to identify the attack surface in the skills. We want to answer the following questions: (1) How many voice commands are sensitive in the current US store (2019)? (2) What kind of security impact it has? (3) Currently, which categories of skills are more sensitive? (4) Do Amazon Alexa and Google Home now perform differently in the security analysis? (5) In the US, how rapidly sensitive voice commands are increasing both in Amazon Alexa and Google Home compare to last year? (6) Do the sensitive skills in one region also appear in the other region?

Sensitive voice commands. To answer the first two questions, we combine our capability analysis (Section 8.4.4) and sensitivity analysis (Section 8.4.5). Through the capability analysis, in the 2019 US store, we find 29,062 action injection and 51,067 information retrieval voice commands from Amazon Alexa, and 3,689 action injection and 5,407 information retrieval voice commands from Google Home. Through our sensitivity analysis, we find that certain skills collect user personal information, banking information, and operate a smart device, smart car. We identified 12,454 security-sensitive voice commands in total (11,442 from Amazon Alexa and 1,012 from Google Home). Putting these analyses together, we show the detailed distributions of sensitive-injection and sensitive-retrieval voice commands of Amazon Alexa US 2019 and Google Home 2019 in Table 8.5 & 8.6 respectively.

Critical categories. We find that several categories contain security-sensitive voice commands than others. From Table 8.5 and 8.6, we can observe that smart home from Amazon Alexa, and home control category from Google Home are the most critical categories respectively. One counter-intuitive result is that several categories (e.g., health & fitness,
8.5. Measurements and Evaluations

<table>
<thead>
<tr>
<th>Plat.</th>
<th>Sensitive</th>
<th>Nonsensitive</th>
</tr>
</thead>
<tbody>
<tr>
<td>US 2019</td>
<td>8,503(10.61%)</td>
<td>2,939(3.67%)</td>
</tr>
<tr>
<td>UK 2019</td>
<td>3,397(6.54%)</td>
<td>757(1.46%)</td>
</tr>
<tr>
<td>US 2018</td>
<td>6,126(9.26%)</td>
<td>1,180(1.78%)</td>
</tr>
<tr>
<td>Good. 2019</td>
<td>671(7.38%)</td>
<td>341(3.74%)</td>
</tr>
<tr>
<td>Good. 2018</td>
<td>566(12.45%)</td>
<td>135(2.97%)</td>
</tr>
</tbody>
</table>

Table 8.4: Overview of the total number of action injection-sensitive, action injection-non sensitive, information retrieval-sensitive, information retrieval-non sensitive voice commands in Amazon Alexa and Google Home.

![Diagram](image)

(a) Comparisons between sensitive skills in Alexa US store from 2018 and 2019

(b) Comparisons between sensitive skills in Google Home from 2018 and 2019

Figure 8.4: Comparisons of Sensitive skills and voice commands between 2018 and 2019 in Amazon Alexa and Google Home.

kids) that seem to be sensitive include less sensitive commands. Intuitively, the health & fitness category contains all the sensitive information such as user health information, daily fitness activity. However, despite those skills, health & fitness category contains lots of skills that give users exercise guidelines or provide other nonsensitive functions. Therefore, the number of nonsensitive commands are higher in this category.

Cross-platform Analysis. We have identified three interesting findings regarding the skills and voice commands in both platforms (Amazon Alexa US 2019 & Google Home 2019). First, we find that there are 42 common sensitive voice commands (such as arm my system, open door one, etc.) in Google Home 2019 and Amazon Alexa US 2019. Moreover, we find 62 vulnerable common skills (such as K Smart, Mysa Thermostat, Lux Products, etc.). Second, we can observe the presence of certain sensitive keywords in most of the sensitive voice commands in both platforms, e.g., Set, Camera, Arm, Add, Check, Lock, Send. Third, for the same category of voice commands, the two platforms have similar percentages of sensitive voice commands in both platforms. For example, smart home in Amazon Alexa has around 12.87% of sensitive voice commands whereas the number is 10.14% for Google home. Analysis on the rest of the categories is included in Table 8.5 & Table 8.6.
Evolution analysis. To understand the trend of voice commands and skills, we compare data between 2019 and 2018 for both platforms (Alexa US & Google Home). We list the detailed result in Table 8.4. First, we find that during 2018-2019, the number of sensitive skills grow from 1,592 (1,385 in Alexa, 207 in Google) to 2,330 (2,144 in Alexa, 186 in Google), and the number of sensitive voice commands grow from 8,007 (7,306 in Alexa, 701 in Google) to 12,454 (11,442 in Alexa, 1,012 in Google). Second, we find that Amazon Alexa has many more newly added sensitive skills and voice commands in 2019. Amazon Alexa has 881 new sensitive skills, and 5,116 new sensitive voice commands, whereas Google Home only has 26 new sensitive skills, and 972 new sensitive voice commands. More importantly, the ratio of sensitive voice commands in Amazon Alexa increases from 11.04% to 14.28%, but in Google Home, this ratio decreases from 15.41% to 11.13%. Third, we noticed 8,228 skills (7,789 from Alexa, and 439 from Google) were taken down. Interestingly, only 2.05% (169 out of 8,228) of the skills are sensitive, which is lower than the percentage of the sensitive skills that remain on the market (5.08%, 1,423 out of 27,996). However, the removed skills contain slightly more sensitive voice commands on average. On average, each removed skill has 5.42 sensitive commands, while the skills on the market have 5.3.

Region-based analysis. To figure out whether voice platforms have different sensitive applications in different regions, besides the US store, we also investigate skills from the UK store in 2019. Interestingly, we have not found any separate store for Google Home other than the US store. As a result, we only compare the UK and the US store for Amazon Alexa. As is shown in Table 8.4, the percentage of sensitive voice skills (3.33%, 674 out of 20,213) is slightly lower than the US. We also found that similar to the US store, the smart home category contains the most sensitive voice commands (3,916 sensitive ones) compared to other categories in the UK store. In addition, we found 11,548 common skills in Alexa US 2019 and Alexa UK 2019 data, and 5.6% (646 out of 11,548) are sensitive skills. Finally, we noticed that 28 sensitive skills only from the UK store did not appear in the US store, while 1,498 sensitive skills only from the US store.

8.5.4 Sanity check evaluation

Only 2% skills (5 out of 250) that we investigate have hidden commands. Therefore, we believe undocumented voice commands are negligible. We identify 3 skills with hidden commands in the communication category, and 2 such skills in the shopping categories, and no skills with hidden commands in the following three categories: productivity, business & finance, and health.
8.6 Discussion

Countermeasure. We find that currently in the US store, 6.74% skills and 13.95% voice commands from Amazon Alexa and Google Home are sensitive. As a result, manufacturers (i.e., Amazon or Google) can introduce an extra layer of protection by authenticating through PIN code or voice profiling before opening sensitive skills. They do not need to impose these restrictions for all the skills. As a result, it will also reduce the overhead of the user while using the nonsensitive skills and ensures a good user experience.

Result Summary and Implications. We perform a large-scale empirical measurement of 82,770 skills and 211,843 voice commands on two popular VPAs – Alexa, and Google Home. We only identify a small portion (5.55%) of the skills that contain sensitive voice commands. Among the sensitive voice commands, there are 19,263 sensitive “action injection” voice commands for controlling smart devices and setting or updating system values. In addition, there are 5,352 sensitive “information retrieval” voice commands for collecting sensitive information about users and the network-connected devices. Across the two VPA platforms (Alexa US 2019 & Google 2019), we only find 62 common sensitive applications available on both platforms and 42 common sensitive voice commands. The results indicate that only a small portion of skills are used for security and privacy-sensitive tasks, which deserves more research.

Limitations. This chapter has a few limitations that need to be further discussed. First, for the sensitivity analysis, we use keyword-based methods, and focus on generating keywords from skills that require account linking. This could lead to false positive cases for the following two reasons: (1) voice commands might include keywords that we did not analyze; (2) the sensitive keywords might be biased because they might not be representative of skills without account linking. However, based on the evaluation of the 275 validation dataset, the false positive rate is only 1.09%. In addition, we manually analyze 270 additional voice commands from 100 randomly selected skills without account linking features, we only identify four sensitive voice commands (false positive rate is 1.48%). A second limitation is related to how we label sensitive voice commands, given that the definition of sensitivity can be quite subjective. In this chapter, we have three people to label each voice command independently, and the agreement rate is reasonably high and acceptable [129] (agreement rate = 94.46%, Fleiss’s Kappa = 0.93). Also, we fine-tuned our sensitive keyword set using a user survey, which would make it well representative. Third, our user study is done through MTurk, and the samples might not be representative of the general population. However, researchers have been using MTurk in prior security & privacy research [121, 141, 221]. Moreover, researchers identified Mechanical Turk as a valid source of high-quality human subject data [174]. Given that, the consistency of reported data, demographically-different samples, we believe our study provides important insights in sensitive voice commands.
Chapter 8. Sensitive Applications of Voice Personal Assistant Systems

8.7 Related Work

Attacks and Defenses in VPAs. Recent research has proved the existence of vulnerabilities in voice interface both in voice controlled system [92, 162, 171, 219, 301, 302] and smartphones [110, 161, 166]. Roy et al. demonstrated an inaudible voice command attack to hijack a user’s VPA [254]. Similarly, dolphin attack [316], Cocaine noodles [281], hidden voice command attack [93] also used inaudible or adversarial voice command to attack VPA. Recently, researchers showed that malicious voice command can also be embedded into audio signal [185, 312]. Apart from these voice commands attacks, attackers are making the VPA system fool by publishing semantically similar malicious skill. Kumar et al. and Zhang et al. demonstrated an innovative way of stealing important user information by skill squatting attack [182, 320]. These NLP level attacks demonstrate serious logic errors and indicate that the speech recognition system is still not mature that makes this system more vulnerable to attack.

Existing defenses include differentiating between human and machine voice, live user detection, voice authentication to protect VPA from attackers [66, 78, 126]. VoiceGesture [319] detected the presence of the live user by extracting user-specific features in the Doppler shift. Uzun et al. used captcha for authenticating the user whenever the system receives a voice command [280]. Researchers analyzed user interaction with the voice assistants [74, 95, 130, 135, 191], efficient way of controlling IoT devices [90, 160]. However, none of them measured the prevalence of sensitive voice commands.

NLP-based Security Analysis. Previous works used NLP techniques to conduct security analysis under various situations such as mobile apps [222, 232, 233], malware [268], privacy policy [200, 255]. For example, LipFuzzer to systematically study the problem of misinterpretation of voice command in VPA systems [317]. While they focused on the pronunciation and functionality of voice commands, our study focuses on capability and sensitivity of commands.

8.8 Conclusion

In this chapter, we measure how a sensitive voice command can affect the security & privacy of VPA. We design an NLP-based tool to analyze sensitive voice command for their security and privacy implications. In our study, we demonstrate the presence of 12,454 sensitive voice commands in the current US store of Amazon Alexa and Google Home, and measure the evolution and region differences.
Table 8.5: We use our two models- active learning model & keyword-based model, to identify the total number of action injection-sensitive, action injection-nonsensitive, information retrieval-sensitive, information retrieval-nonsensitive voice commands in 80,129 voice commands from twenty-three different categories of Amazon Alexa US 2019. Inject. means action injection. Retriv. means information retrieval.
Table 8.6: Analysis results for 9,096 voice commands from eighteen different categories of Google Home 2019.

<table>
<thead>
<tr>
<th>Category</th>
<th>Sensitive</th>
<th>Nonsensitive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Home Control</td>
<td>642(24.51%) Inject.</td>
<td>1,092(41.7%) Inject.</td>
</tr>
<tr>
<td></td>
<td>1,092(41.7%) Retriv.</td>
<td>604(23.06%) Retriv.</td>
</tr>
<tr>
<td>Productivity</td>
<td>10(3.45%) Inject.</td>
<td>119(41.03%) Inject.</td>
</tr>
<tr>
<td></td>
<td>14(4.83%) Retriv.</td>
<td>147(50.69%) Retriv.</td>
</tr>
<tr>
<td>Shopping</td>
<td>6(1.22%) Inject.</td>
<td>135(27.44%) Inject.</td>
</tr>
<tr>
<td></td>
<td>20(4.07%) Retriv.</td>
<td>331(67.28%) Retriv.</td>
</tr>
<tr>
<td>Health &amp; Fit.</td>
<td>4(0.94%) Inject.</td>
<td>163(38.44%) Inject.</td>
</tr>
<tr>
<td></td>
<td>- Retriv.</td>
<td>257(60.61%) Retriv.</td>
</tr>
<tr>
<td>Communication</td>
<td>4(1.67%) Inject.</td>
<td>64(26.67%) Inject.</td>
</tr>
<tr>
<td></td>
<td>- Retriv.</td>
<td>172(71.67%) Retriv.</td>
</tr>
<tr>
<td>Movies &amp; TV</td>
<td>3(1.29%) Inject.</td>
<td>99(42.67%) Inject.</td>
</tr>
<tr>
<td></td>
<td>- Retriv.</td>
<td>130(56.03%) Retriv.</td>
</tr>
<tr>
<td>Trvl &amp; Trans.</td>
<td>1(0.23%) Inject.</td>
<td>96(22.33%) Inject.</td>
</tr>
<tr>
<td></td>
<td>6(1.4%) Retriv.</td>
<td>327(76.05%) Retriv.</td>
</tr>
<tr>
<td>Food &amp; Drink</td>
<td>1(0.43%) Inject.</td>
<td>66(28.33%) Inject.</td>
</tr>
<tr>
<td></td>
<td>3(1.29%) Retriv.</td>
<td>163(69.96%) Retriv.</td>
</tr>
<tr>
<td>Busin. &amp; Fin.</td>
<td>- Inject.</td>
<td>29(7.95%) Inject.</td>
</tr>
<tr>
<td></td>
<td>11(3.01%) Retriv.</td>
<td>325(89.04%) Retriv.</td>
</tr>
<tr>
<td>Edu. &amp; Ref.</td>
<td>- Inject.</td>
<td>104(13.87%) Inject.</td>
</tr>
<tr>
<td></td>
<td>3(0.4%) Retriv.</td>
<td>643(85.73%) Retriv.</td>
</tr>
<tr>
<td>News</td>
<td>- Inject.</td>
<td>416(44.44%) Inject.</td>
</tr>
<tr>
<td></td>
<td>2(0.21%) Retriv.</td>
<td>518(55.34%) Retriv.</td>
</tr>
<tr>
<td>Local</td>
<td>- Inject.</td>
<td>26(8.78%) Inject.</td>
</tr>
<tr>
<td></td>
<td>1(0.34%) Retriv.</td>
<td>269(90.88%) Retriv.</td>
</tr>
<tr>
<td>Music &amp; Aud.</td>
<td>- Inject.</td>
<td>246(80.13%) Inject.</td>
</tr>
<tr>
<td></td>
<td>- Retriv.</td>
<td>61(19.87%) Retriv.</td>
</tr>
<tr>
<td>Games &amp; Triv.</td>
<td>- Inject.</td>
<td>208(25.74%) Inject.</td>
</tr>
<tr>
<td></td>
<td>- Retriv.</td>
<td>600(74.26%) Retriv.</td>
</tr>
<tr>
<td>Sports</td>
<td>- Inject.</td>
<td>57(32.39%) Inject.</td>
</tr>
<tr>
<td></td>
<td>- Retriv.</td>
<td>119(67.61%) Retriv.</td>
</tr>
<tr>
<td>Weather</td>
<td>- Inject.</td>
<td>50(30.3%) Inject.</td>
</tr>
<tr>
<td></td>
<td>- Retriv.</td>
<td>115(69.7%) Retriv.</td>
</tr>
<tr>
<td>Art &amp; Life.</td>
<td>- Inject.</td>
<td>34(13.08%) Inject.</td>
</tr>
<tr>
<td></td>
<td>- Retriv.</td>
<td>226(86.92%) Retriv.</td>
</tr>
<tr>
<td>Kids</td>
<td>- Inject.</td>
<td>14(19.18%) Inject.</td>
</tr>
<tr>
<td></td>
<td>- Retriv.</td>
<td>59(80.82%) Retriv.</td>
</tr>
</tbody>
</table>
Chapter 9

Conclusions and Future Works

9.1 Conclusion

The goal of this thesis is to characterize online deception and efficiently detect it. In this thesis, we have covered the full cycle of the most typical online phishing process.

With email system, we conducted end-to-end measurements and real-world phishing tests. We show that most email providers allow forged emails into user inbox. Most service providers still lack the necessary warning mechanisms. For the few email services that implemented security indicators, we show that security indicators have a positive impact on reducing risky user actions under phishing attacks but cannot eliminate the risk. We hope the results can help to draw more community attention to promoting the adoption of SMTP security extensions, and developing effective security indicators for the web and mobile email interfaces.

We followed up with a survey with email administrators by asking why the adoption rates of anti-spoofing protocols are low. By analyzing the discussion threads in IETF and performing user studies with email administrators, we provide a deeper understanding of the perceived value and limitations of anti-spoofing protocols. Our results show that key security and usability limitations are rooted in the protocol design which hurts the perceived usefulness of these protocols. This also makes it difficult to establish a “critical mass” to facilitate a positive net effect for a wider adoption. Moving forward, extensive efforts are needed to address the technical issues in the protocol design and develop external enforcement (or incentives) to bootstrap the protocol adoption.

In this thesis, we also perform a first measurement study on disposable email services. We collect a large dataset from 7 popular disposable email services (2.3 million emails sent by 210K domains), and provide new understandings of what disposable email services are used for and the potential risks of usage. In addition, we use the collected email dataset to empirically analyze email tracking activities. Our results provide new insights into the prevalence of tracking at different online services and the evasive tracking methods used of trackers. The results are valuable for developing more effective anti-tracking tools.

Moving on to phishing websites, we perform an extensive measurement on squatting phishing, where the phishing pages impersonate target brands at both the domain and content level. By monitoring 700+ brands and 600K squatting domains for a month, we identified 857
phishing web pages and 908 mobile pages. We show that squatting phishing pages are impersonating trusted entities through all different domain squatting techniques. Squatting phishing pages are more likely to adopt evasion techniques and are hard to catch. About 90% of them have evaded the detection of popular blacklists for at least a month.

To better understand credentials stolen on phishing websites, we perform an empirical measurement on the information flows of credential sharing during phishing attacks. Our analysis covers more than 179,000 phishing URLs (47,000 live phishing sites). We show that user credentials are shared in real-time to multiple parties at both the client side and the server side. Although third-party sharing exposes user credentials to even more malicious parties, we argue that defenders may make use of these channels to back-track phishing servers and alert phishing victims.

With the emergence of voice personal assistant, it becomes a new opportunity of online deception. We measure how a sensitive voice command can affect the security & privacy of VPA. We design an NLP-based tool to analyze sensitive voice command for their security and privacy implications. In our study, we demonstrate the presence of 12,454 sensitive voice commands in the current US store of Amazon Alexa and Google Home, and measure the evolution and region differences.

**9.2 Future Works**

There are still open questions we find during our researches in the field of online deception. Here, I want to talk about a few future directions that I think can have great impact. First, most of phishing researches focus on phishing detection or measurements. However, the user education of techniques used in phishing including email impersonation, domain impersonation and emotion manipulation are rarely discussed. While phishers exploiting users emotion and weakness, users should be improved in the cycle of improving overall system security. Therefore, effective ways to educate users of potential attacks including phishing are worth exploring. Second, with the characterization of email tracking services, it’s straightforward to develop more effective email tracking blocking services. The tracking blocker should be transparent to users and doesn’t sacrifice users’ experience. Third, previous deception attacks use fake emails or fake websites to gain trust. However, with the advancement of DeepFake, it becomes possible that attackers can use fake audio or fake video to gain trust. However, it isn’t clear that what are the challenges of using DeepFake to deceive people, and what are the available countermeasures. It’s worthwhile to explore in this direction.

With large data set collected and analyzed, this thesis characterizes and detects online deception.
Bibliography


[34] Puppeteer: Headless chrome node api. https://github.com/GoogleChrome/puppeteer/.


[41] Cyber squatters are targeting britain’s biggest banks. https://goo.gl/.


[189] Frederic Lardinois. Gmail now has more than 1b monthly active users. Tech Crunch, 2016. https://techcrunch.com/2016/02/01/gmail-now-has-more-than-1b-monthly-active-users/.


Appendices
A Spoofing Target Domains

Table A.1 lists the 30 domains used by the end-to-end spoofing experiment as the spoofed sender address. The domains per category are selected randomly from Alexa top 5000 domains.

Table A.1: Spoofed Sender Domain List.

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
<th>Domains</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>No SPF/DKIM/DMARC (10)</td>
<td>thepiratebay.org, torrent-baza.net, frdic.com, chinafloor.cn, onlinesbi.com, 4dsplay.com, peliculasflv.tv, sh.st, contw.com, anyanime.com</td>
</tr>
<tr>
<td>Relaxed</td>
<td>SPF/DKIM;DMARC=none (10)</td>
<td>tumblr.com, wikipedia.org, ebay.com, microsoftonline.com, msn.com, apple.com, vt.edu, github.com, qq.com, live.com</td>
</tr>
<tr>
<td>Strict</td>
<td>SPF/DKIM;DMARC=reject (10)</td>
<td>google.com, youtube.com, yahoo.com, vk.com, reddit.com, facebook.com, twitter.com, instagram.com, linkedin.com, blogspot.com</td>
</tr>
</tbody>
</table>

Other Vulnerabilities. We find that 2 email services “sapo.pt” and “runbox.com” are not carefully configured, allowing an attacker to piggyback on their mail servers to send forge emails. This threat model is very different from our experiments above, and we briefly describe it using Figure 2.1. Here, the attacker is the sender MUA, and the vulnerable server (e.g., runbox.com) is the sender service. Typically, Runbox should only allow its users to send an email with the sender address as “{someone}@runbox.com”. However, the Runbox’s server allows a user (the attacker) to set the “MAIL FROM” freely (without requiring a verification) in step 1 to send forged emails. This attack does not help the forged email to bypass the SPF/DKIM check. However, it gives the attacker a static and reputable IP address. If the attacker aggressively sends malicious emails through the vulnerable mail server, it can damage the reputation of the IP. We have reported the vulnerability to the service admins.

Misleading User Interface. Figure A.1 shows three examples of misleading UI elements. Figure A.1a and A.1b show that when an attacker spoofs a user from the same email provider as the receiver, the email provider will automatically load the profile photo of the spoofed sender from its internal database. In both Google Inbox and Seznam, the forged emails look like that they were sent by the user “Forged”, and the photo icon gives the forged email a more authentic look. Figure A.1c demonstrates the misleading UIs when the attacker spoofs an existing contact of the receiver. Again, despite the sender address (contact@vt.edu) is spoofed, Zoho still loads the contact’s photo from its internal database. In addition, users can check the recent email conversations with this contact by clicking on the highlighted link. These elements make the forged email look authentic.
Hash or encoding functions (31 in total)
MD2, MD4, MD5, RIPEMD, SHA1, SHA224, SHA256, SHA384,
SHA512, SHA3_224, SHA3_256, SHA3_384, SHA3_512, blake2b,
blake2s, crc32, adler32, murmurhash 3 32 bit, murmurhash 3 64 bit,
murmurhash 3 128 bit, whirlpool, b16 encoding, b32 encoding,
b64 encoding, b85 encoding, url encoding, gzip, zlib, bz2, yenc, entity

Table B.1: Functions to obfuscate user identifiers.

B Obfuscated User Identifier

To detect obfuscated user identifiers (i.e. email addresses) in the tracking URLs, we have
tested 31 different hash/encoding functions. If the link’s parameters contain the “obfuscated
version” of the receiver’s email address, then the image is considered as a tracking pixel.
As shown in Table B.1, we apply 31 hash/encoding functions on the receiver email address
to look for a match. We also test two-layer obfuscations by exhaustively applying two-
function combinations, e.g., \texttt{MD5(SHA1())}. In total, we examine 992 obfuscated strings for
each address.

C Example of Sensitive and Non-Sensitive Voice Commands.

We have listed several sensitive voice commands from Amazon Alexa and Google Home in
Table C.2 and Table C.1, respectively.

<table>
<thead>
<tr>
<th>Category</th>
<th>Command</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Home Control</td>
<td>“dim the lights”</td>
<td>Injection</td>
</tr>
<tr>
<td></td>
<td>“turn off the lights”</td>
<td>Injection</td>
</tr>
<tr>
<td></td>
<td>“turn down the thermostat”</td>
<td>Injection</td>
</tr>
<tr>
<td></td>
<td>“are the lights on in the kids bedroom”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“set the thermostat to 68 degrees”</td>
<td>Injection</td>
</tr>
<tr>
<td></td>
<td>“show garage camera”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“whats on baby camera”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“show the front door camera on living room”</td>
<td>Retrieval</td>
</tr>
<tr>
<td>Productivity</td>
<td>“find my keys”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“turn on climate control for my car”</td>
<td>Injection</td>
</tr>
<tr>
<td></td>
<td>“arm stay”</td>
<td>Injection</td>
</tr>
<tr>
<td></td>
<td>“set scene to away”</td>
<td>Injection</td>
</tr>
<tr>
<td></td>
<td>“lock doors”</td>
<td>Injection</td>
</tr>
<tr>
<td></td>
<td>“record video”</td>
<td>Injection</td>
</tr>
</tbody>
</table>
### C. Example of Sensitive and Non-Sensitive Voice Commands.

<table>
<thead>
<tr>
<th>Category</th>
<th>Command</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smart Home</td>
<td>“turn on the sprinklers”</td>
<td>Injection</td>
</tr>
<tr>
<td></td>
<td>“turn on the defroster”</td>
<td>Injection</td>
</tr>
<tr>
<td>Shopping</td>
<td>“about my order”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“add my usuals”</td>
<td>Injection</td>
</tr>
<tr>
<td>Kids</td>
<td>“add feeding thirty five ounces”</td>
<td>Injection</td>
</tr>
<tr>
<td></td>
<td>“add poop and pee”</td>
<td>Injection</td>
</tr>
<tr>
<td>Art &amp; Lifestyle</td>
<td>“find my phone”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“turn on the bedroom lamp”</td>
<td>Injection</td>
</tr>
<tr>
<td></td>
<td>“ring my phone”</td>
<td>Injection</td>
</tr>
<tr>
<td></td>
<td>“the location of my wallet”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“the location of my keys”</td>
<td>Retrieval</td>
</tr>
<tr>
<td>Business &amp; Finance</td>
<td>“what is my account balance”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“check my savings account”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“list my wallets”</td>
<td>Retrieval</td>
</tr>
<tr>
<td>Connected Car</td>
<td>“where is my car”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“information about my last trip”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“unlock my car with pin 1234”</td>
<td>Injection</td>
</tr>
<tr>
<td></td>
<td>“the location of my car”</td>
<td>Retrieval</td>
</tr>
</tbody>
</table>

Table C.1: Sensitive commands from Google Home.
<table>
<thead>
<tr>
<th>Sensitive commands from Amazon Alexa.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Lock</strong></td>
</tr>
<tr>
<td>“lock my car” Injection</td>
</tr>
<tr>
<td>“how much gas i have” Retrieval</td>
</tr>
<tr>
<td>“do i need gas” Retrieval</td>
</tr>
<tr>
<td>“start my car” Injection</td>
</tr>
<tr>
<td>“check my battery status” Retrieval</td>
</tr>
<tr>
<td><strong>Food &amp; Drink</strong></td>
</tr>
<tr>
<td>“order food” Injection</td>
</tr>
<tr>
<td>“reorder my last order” Injection</td>
</tr>
<tr>
<td>“add avocados to my order” Injection</td>
</tr>
<tr>
<td>“track my order” Retrieval</td>
</tr>
<tr>
<td>“add me to the waitlist” Injection</td>
</tr>
<tr>
<td>“order my most recent order” Injection</td>
</tr>
<tr>
<td>“check my balance” Retrieval</td>
</tr>
<tr>
<td>“add the expired items to my shopping” Injection</td>
</tr>
<tr>
<td><strong>Health &amp; Fitness</strong></td>
</tr>
<tr>
<td>“order replacement contact lenses” Injection</td>
</tr>
<tr>
<td>“what is my copay for a hospital stay” Retrieval</td>
</tr>
<tr>
<td>“add a symptom” Injection</td>
</tr>
<tr>
<td>“cancel the call” Injection</td>
</tr>
<tr>
<td>“send help” Injection</td>
</tr>
<tr>
<td>“track a seizure” Retrieval</td>
</tr>
<tr>
<td>“add a blood pressure reading” Injection</td>
</tr>
<tr>
<td>“record 150 pounds” Injection</td>
</tr>
<tr>
<td><strong>Shopping</strong></td>
</tr>
<tr>
<td>“where is my order” Retrieval</td>
</tr>
<tr>
<td>“i want to return an item” Injection</td>
</tr>
<tr>
<td>“add apples to my shopping list” Injection</td>
</tr>
<tr>
<td>“what do i have on my shopping list” Retrieval</td>
</tr>
<tr>
<td>“put apples on the list” Injection</td>
</tr>
<tr>
<td>“text me the list of items” Injection</td>
</tr>
<tr>
<td>“add cheese to my list” Injection</td>
</tr>
<tr>
<td>“what is on my list” Retrieval</td>
</tr>
<tr>
<td>“add carrots to my shopping list” Injection</td>
</tr>
<tr>
<td>“when my order will arrive” Retrieval</td>
</tr>
</tbody>
</table>
We have listed a few non-sensitive voice commands from Amazon Alexa and Google Home in Table C.3 and Table C.4.

<table>
<thead>
<tr>
<th>Category</th>
<th>Command</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Movies &amp; Tv</strong></td>
<td>“give me a fact”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“get the rating of the big bang theory”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“give me the upcoming movies”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“whats my flash briefing”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“when is the next episode of game of thrones”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“what are the new releases on netflix”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“give me a good movie suggestion”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“hat is the score of star wars”</td>
<td>Retrieval</td>
</tr>
<tr>
<td><strong>Travel &amp; Transp.</strong></td>
<td>“top five things to do”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“tell me some facts about arlington”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“if is it safe tomorrow on california street”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“get status for the port jefferson”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“how far away is it to yosemite from santa barbara”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“are there any alerts for the needham line”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“where to ride today”</td>
<td>Retrieval</td>
</tr>
<tr>
<td><strong>Games, Trivia &amp; Access.</strong></td>
<td>“give me a shark fact”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“the damage of the alien engine”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“tell me some facts about orlando”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“guide for a quote”</td>
<td>Retrieval</td>
</tr>
<tr>
<td><strong>Shopping</strong></td>
<td>“what today’s specials are”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“about the deal of the day”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“what garage sales are coming up”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“how much is the ford escape”</td>
<td>Retrieval</td>
</tr>
<tr>
<td><strong>Lifestyle</strong></td>
<td>“who is performing this saturday”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“what are my upcoming band events”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“whats the hebrew date today”</td>
<td>Retrieval</td>
</tr>
<tr>
<td></td>
<td>“what goes with oversize sweater”</td>
<td>Retrieval</td>
</tr>
</tbody>
</table>

Table C.3: Non-sensitive commands from Amazon Alexa.
<table>
<thead>
<tr>
<th>Category</th>
<th>Command</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local</td>
<td>“prayers time in chicago”</td>
<td>Retrieval</td>
</tr>
<tr>
<td>Local</td>
<td>“what is the definition of leather”</td>
<td>Retrieval</td>
</tr>
<tr>
<td>Local</td>
<td>“what is the second verse of the star spangled banner”</td>
<td>Retrieval</td>
</tr>
<tr>
<td>Local</td>
<td>“travel conditions in marion county”</td>
<td>Retrieval</td>
</tr>
</tbody>
</table>
### C. Example of Sensitive and Non-Sensitive Voice Commands.

<table>
<thead>
<tr>
<th>Table C.4: Non-sensitive commands from Google Home.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Movies &amp; Tv</strong></td>
</tr>
<tr>
<td>“can i recycle an egg carton”</td>
</tr>
<tr>
<td>“get me a plumber in brighton”</td>
</tr>
<tr>
<td>“give me details for the department of corrections”</td>
</tr>
<tr>
<td>“why do i need a will”</td>
</tr>
<tr>
<td><strong>Travel &amp; Transp.</strong></td>
</tr>
<tr>
<td>“a good movie to watch”</td>
</tr>
<tr>
<td>“recommend a movie”</td>
</tr>
<tr>
<td>“what is new on amazon prime video”</td>
</tr>
<tr>
<td>“when was cosmopolitan established”</td>
</tr>
<tr>
<td>“when was the first appearance of spiderman”</td>
</tr>
<tr>
<td>“who plays mickey mouse”</td>
</tr>
<tr>
<td><strong>Games, Trivia &amp; Access.</strong></td>
</tr>
<tr>
<td>“when does next the bart train leave”</td>
</tr>
<tr>
<td>“what are the places to visit in san francisco”</td>
</tr>
<tr>
<td>“what are the rides at universal studios hollywood”</td>
</tr>
<tr>
<td>“how many square miles is santorini”</td>
</tr>
<tr>
<td>“what is the population of rome”</td>
</tr>
<tr>
<td>“how long is the carnival magic”</td>
</tr>
<tr>
<td><strong>Shopping</strong></td>
</tr>
<tr>
<td>“why kyle maclachlan cares about tigers”</td>
</tr>
<tr>
<td>“tell me a physics joke”</td>
</tr>
<tr>
<td>“calculate kinetic energy”</td>
</tr>
<tr>
<td>“tell me something funny”</td>
</tr>
<tr>
<td><strong>Arts &amp; Lifestyle</strong></td>
</tr>
<tr>
<td>“what the real estate market is doing”</td>
</tr>
<tr>
<td>“what is the closest dry cleaner”</td>
</tr>
<tr>
<td>“find a brunch recipe”</td>
</tr>
<tr>
<td>“recommend strong coffee beans”</td>
</tr>
<tr>
<td>“search for beer in berlin”</td>
</tr>
<tr>
<td>“who wrote the marriage of figaro”</td>
</tr>
<tr>
<td>“tell me the currency of japan”</td>
</tr>
<tr>
<td>“give me a catholic fact”</td>
</tr>
<tr>
<td>“how tall is ellen degeneres”</td>
</tr>
</tbody>
</table>

*Retrieval*
Figure A.1: Examples of misleading UIs (profile photo, email history, namecard).