# Leveraging Scalable Data Analysis to Proactively Bolster the Anti-Phishing Ecosystem

by

Adam Oest

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved April 2020 by the
Graduate Supervisory Committee:

Gail-Joon Ahn, Co-Chair
Adam Doupé, Co-Chair
Yan Shoshitaishvili
RC Johnson

ARIZONA STATE UNIVERSITY

May 2020

ABSTRACT

Despite an abundance of defenses that work to protect Internet users from online threats, malicious actors continue to deploy relentless large-scale phishing attacks that target these users. Effectively mitigating phishing attacks remains a challenge for the security community due to attackers' ability to evolve and adapt to defenses, the cross-organizational nature of the infrastructure abused for phishing, and discrepancies between theoretical and realistic anti-phishing systems. Although technical countermeasures cannot always compensate for the human weakness exploited by social engineers, maintaining a clear and up-to-date understanding of the motivation behind—and execution of—modern phishing attacks is essential to optimizing such countermeasures.

In this dissertation, I analyze the state of the anti-phishing ecosystem and show that phishers use evasion techniques, including *cloaking*, to bypass anti-phishing mitigations in hopes of maximizing the return-on-investment of their attacks. I develop three novel, scalable data-collection and analysis frameworks to pinpoint the ecosystem vulnerabilities that sophisticated phishing websites exploit. The frameworks, which operate on real-world data and are designed for continuous deployment by anti-phishing organizations, empirically measure the robustness of industry-standard anti-phishing blacklists (*PhishFarm* and *PhishTime*) and proactively detect and map phishing attacks prior to launch (*Golden Hour*). Using these frameworks, I conduct a longitudinal study of blacklist performance and the first large-scale end-to-end analysis of phishing attacks (from spamming through monetization). As a result, I thoroughly characterize modern phishing websites and identify desirable characteristics for enhanced anti-phishing systems, such as more reliable methods for the ecosystem to collectively detect phishing websites and meaningfully share the corresponding intelligence. In addition, findings from these studies led to actionable security recommendations that were implemented by key organizations within the ecosystem to help improve the security of Internet users worldwide.

*To Tram, whose unwavering dedication helped me along every step of my Ph.D. journey.*

ACKNOWLEDGMENTS

My research would not have been possible without the tremendous support of my advisors, fellow students, and industry collaborators.

Dr. Ahn, I still remember when I came into your office on the first day of my Ph.D. to select a research project. Little did I know that the very same project would inspire much of the work in this dissertation. Thanks to you, I was able to hit the ground running and acquired a wealth of new knowledge in a very short time. I truly appreciate the guidance you've given me to help make my work successful.

Dr. Doupé, your Software Security class was eye-opening and gave me the foundation I needed to be successful in much of my Ph.D., in part by showing the far-reaching impact that security research can have. I also have fond memories of the *Project CTF*. I'd like to thank you for all the help and advice you've given me with my projects and papers over the past several years, which has been key to my growth as a researcher.

Dr. Shoshitaishvili, thanks for introducing me to the fascinating world of binary analysis, which has given me a new perspective on research that I otherwise probably wouldn't have considered. I'm eager to one day be able to put the knowledge I've learned in this area to good use. I'd also like to thank you for serving on my committee, giving me insightful feedback along the way, and always having a cheerful attitude.

Dr. Johnson, thank you so much for your continuous support ever since you joined my committee, which was especially helpful in the final months of my Ph.D. I really appreciate your excitement about my work and your passion to bridge academic research with real-world impact. I look forward to what lies ahead.

Dr. Wardman, I've been impressed by your ability to identify so many intriguing research problems relevant to the current state of web security. Thank you for opening new doors for me, for making it possible to be a part of solving some of these problems, and for your enthusiasm about research.

It has been an honor to be a part of the SEFCOM lab, and I'd like to thank all my co-authors and fellow students who have worked with me on projects over the past years. I'd also like to thank the professors for supporting the exciting trips and activities that we've had outside of the lab, and for their dedication to driving the lab's spectacular growth. Thanks to your efforts, my Ph.D. has become more than I could ever imagine. I hope that our paths will cross again in the future.

Last but certainly not least, I want to thank my incredible wife, Tram, for helping me survive the many days (and nights) of tireless work throughout my academic career, and for always bringing happiness and love even in the most stressful of times. In addition, I'd like to thank my parents for steering me in the right direction by prioritizing my education: in particular, my dad is the one who first inspired me to study computer science, and I am grateful for his lifelong encouragement.

My research projects were supported in part by grants from PayPal, Inc. and the Center for Cybersecurity and Digital Forensics at ASU. Parts of the research would not have been possible without the help of Foy Shiver, Peter Cassidy, and others at the Anti-Phishing Working Group.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

CHAPTER Page

CHAPTER Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

# Chapter 1

# INTRODUCTION

Over the course of less than three decades of being available to the public, the Internet has fundamentally transformed the way in which modern society functions. This behemoth resource plays a critical role in areas such as telecommunication, education, entertainment, and commerce. As of December 2019, an estimated 4.6 *billion* people use the Internet—some 58.7% of the world's population [85]. Furthermore, the growth and influence of the Internet show no sign of stopping. In fact, many current Internet users might find it hard to imagine daily life without access to the services powered by the Internet, which accounts for over 99% of all global communications [54].

The origins of the Internet can be traced back to ARPANET, which was a research project led by the Defense Advanced Research Projects Agency (DARPA) from the late 1960s. ARPANET initially interconnected computers at just four different universities, but later led to the formalization of protocols that underpin the modern Internet, such as TCP/IP [108]. It was not long after the launch of ARPANET that it became clear that malicious users could easily gain unauthorized access to parts of the network, and that there existed a "lingering affection for the challenge of breaking someone's system" [83]. As a result, the security of computer systems turned out to be a larger concern than anyone at the time had anticipated.

The invention of the World Wide Web by Tim Berners-Lee in 1989 marked the beginning of the Internet's transformation away from being a resource only for scientists [10]. By 1993, over one million public *hosts* were accessible through the Internet. Growth exploded to 171 million hosts by 2003 and exceeded one billion in late 2013 [60]. Unfortunately, as the scale of the Internet grew, so did the motivation of attackers to exploit secu-

rity vulnerabilities. Today, this motivation is exacerbated by the potential for considerable monetary gain, as it is the norm—rather than the exception—for sensitive or financial information to be transmitted across the Internet and stored on remote systems. Should such information fail to be adequately protected, it could fall into the hands of criminals rather than the intended recipients, and those criminals could subsequently monetize the information at the expense of one or more victims.

## 1.1 Cybercrime

Cybercrime, also known as computer crime or electronic crime, refers to a broad class of criminal activity that involves digital devices (e.g., computers, smartphones, or tablets), computer networks, or the Internet. Like traditional crime, cybercrime ranges in scope from individual attackers, to organized groups, to nation-state actors, including terrorists. Similarly, cybercriminals can use computers as tools to commit a crime, as targets of crime, or both.

Effective prevention of cybercrime presents several unique challenges, as cybercrime is an ever-evolving landscape that rapidly adapts to changes in technology and user trends. The proliferation of connected devices and ease of access to the Internet is a double-edged sword: although the legitimate use of such technology can greatly benefit society, its broad availability also gives criminals a growing attack surface with a diminishing barrier to entry. Simultaneously, cybercriminals have access to a growing suite of anonymization techniques that they use to cover their tracks. On top of this, underground economies provide cybercriminals with access to sophisticated attack tools without the need for advanced knowledge [102].

Cybercrime regularly propagates across jurisdictions, which in turn hampers prosecution by law enforcement entities. Although there have historically been several high-profile convictions for computer-related crimes, the legal consequences for—and laws and

policies against—many types of cybercrime remain insufficient. This legal ambiguity is particularly problematic when trying to counter cybercrime characterized by a very large number of small-scale perpetrators.

The annual damage caused by cybercrime is difficult to definitively measure due to the diversity and worldwide distribution of the crimes committed, and the numerous defenses which are required as a result. However, many researchers agree that the impact of cybercrime has been underestimated by governments and likewise under-reported by victims [39]. A 2017 study by anti-virus vendor McAfee estimated *annual* global damage due to cybercrime to account for $445-608 billion, or nearly 1% of the total GDP of the world's developed economies and a $100 billion increase in just three years [81].

Large-scale incidents directly illustrate the staggering real-world consequences of cybercrime, which can spread beyond the confines of the realm of computers. In 2017, hackers breached Equifax and were able to steal detailed personal information of over 140 million Americans—an unprecedented impact, nearly equivalent to the country's entire labor force. The $1 billion of direct damage suffered by Equifax in the immediate aftermath is dwarfed by long-term consequences such as mass identity theft, the risk of irreversible brand damage, and even espionage [9]. Around the same time, the ransomware known as *WannaCry* infected over 300,000 computers worldwide and subsequently caused tangible damage by disrupting numerous hospitals, vehicle factories, and public transportation networks prior to being contained [88]. As of late 2019, no arrests had been made in either case nor had the perpetrators been directly identified.

On the other hand, everyday cybercrime does not necessarily make headlines, as it often involves small attacks. However, these attacks can occur with such frequency that they are collectively no less significant than high-profile breaches. As one of their most routine attacks, cybercriminals perform tens of billions of vulnerability scans each day to attempt to gain unauthorized access to computer systems across the Internet [81]. Beyond

wasting network bandwidth worldwide, such scans force the owners of computers with network access to always remain vigilant and quickly address vulnerabilities as they are discovered. The necessity of vigilance is a recurring theme in the fight against attackers because individual users are also targeted directly. In 2019, more than half of all e-mail traffic was estimated to be spam (unwanted or abusive), with content ranging from unwanted advertising to dangerous malware [62]. As part of their spam campaigns, attackers flood the Internet with hundreds of new deceptive websites each day to trick victims into making payments or revealing sensitive information [5]. At times, attacks are even delivered through more traditional means such as phone calls or text messages [128]. The FBI Internet Crime Complaint Center reported that the compromise of business e-mail accounts alone caused over $12 billion in total reported monetary loss between 2013 and 2018 [39]. The broad range of attack types suggests that this is just the tip of the iceberg, and thus effectively thwarting the attackers' onslaught is a daunting challenge.

Ultimately, around virtually every corner of today's technologically-heavy world, an attacker stands ready to cause damage to a victim, whether it be in terms of time, money, information, or intellectual property. Fortunately, a wide gamut of defenses has already been proposed to protect victims from many such attackers. However, the burden of effectively implementing these defenses falls upon the individuals, organizations, and governments involved, who must make a concentrated and deliberate effort to address each threat. At the same time, research efforts must keep up with (and, ideally, exceed) attackers' capabilities to ensure mitigations remain sufficient and that a proper understanding of attacks is maintained over time.

## 1.2 Phishing

Phishing—a type of attack in which victims are tricked into disclosing sensitive information by means of social engineering—is a prime example of one type of the afore-

mentioned everyday cybercrime, and it occurs on a mammoth scale. Phishing typically involves *lures* such as deceptive (spam) e-mails or phone calls to catch the attention of victims, who are later directed to a fake (but convincing) web page or application to unsuspectingly submit credentials, disclose personal information, or even send payments. Attackers then use the submitted data for fraudulent purposes, usually with the end goal of monetization.

On the surface, phishing attacks may seem to be quite trivial, as they often consist of just a few straightforward steps such as setting up a website and sending e-mails. One might therefore reasonably assume that mitigating such a simple attack poses no challenge, and that modern technology has surely eliminated phishing to allow for our valuable time to be spent addressing more sophisticated threats.

Indeed, since the early days of the World Wide Web, numerous defenses against phishing have been proposed and implemented [110]. We have moved from an era of heterogeneous, marginally effective browser security toolbars [137] to one with anti-phishing systems that operate on a global scale and are natively available in modern web browsers and e-mail systems [101, 130, 136]. Such systems are further augmented by multi-layer security strategies developed by anti-phishing organizations, such as user awareness training [68] and two-factor authentication [129].

Yet, despite the many efforts of researchers and engineers, not only has the security community failed to thwart phishing, but phishing attacks continue to evolve in terms of both volume and sophistication. Even as of 2019, every month, millions of Internet users are the potential victims of phishing by tens or hundreds of thousands of deceptive websites [6, 78]. Per the Google Safe Browsing Transparency Report, the sustained volume of phishing websites reached an all-time high in 2019, and phishing has almost completely replaced malware as the primary threat against web browsers [47].

Specially-crafted variations of phishing attacks—known as *spearphishing*—have historically also targeted high-profile victims [55]. For example, during the 2016 US presidential election, attackers successfully used a deceptive e-mail to obtain account credentials of a senior campaign manager and subsequently publicly leaked over 20,000 pages of private e-mail communications [49]. Globally, the total annual damage from phishing has been estimated to range up to $3 billion [57][1].

In 2003, the Anti-Phishing Working Group (APWG) was founded as a non-profit association to address the "growing problem of phishing, crimeware, and e-mail spoofing" [3], with membership open to businesses, organizations, and government agencies either affected by phishing or otherwise involved in anti-phishing efforts. Among the APWG's key activities are sharing phishing intelligence and regularly publishing phishing trend reports. In one of its first reports, dated October 2004, the APWG detected some 1,142 unique phishing websites and 6,597 unique phishing e-mails [3]. Today, such statistics are on the order of hundreds of thousands per month [5]. Figure 1.1 summarizes the APWG's aggregate phishing metrics over the course of the past 15 years.

Due to changes in reporting methodologies (and the broader Internet ecosystem) over time, reports such as those of the APWG should not be considered as a definitive summary of long-term phishing trends. However, we can certainly observe a sustained volume of phishing attacks over the years. Moreover, as a percentage of all hostnames on the Internet, the representation of phishing websites more than doubled between 2005 and 2018, from approximately 0.047% to 0.102% [60]. It is reasonable to suspect that phishers are able to maintain a sufficient *return on investment* to justify carrying out (or growing) their attacks, even amid modern defenses.

[1] Accurately quantifying the damage caused by phishing is itself a difficult problem. I discuss the reasons for this in Chapters 2 and 6.

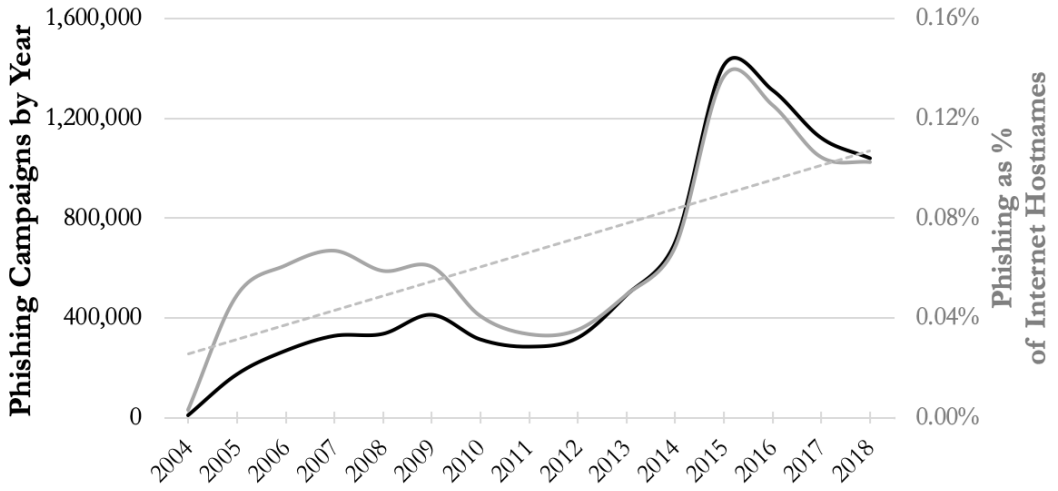



Figure 1.1: Growth of phishing attacks from 2004 to 2018 (APWG and ISC) [5, 60].

Given the aforementioned observations, it should not be surprising that the effective protection of victims from phishing attacks poses a number of difficult challenges—many of which are yet to be solved. Technical countermeasures against phishing are of paramount importance as they have the ability to automatically protect users at scale. However, because phishing attacks function in part by exploiting the human sitting behind the screen, technical mitigations alone will not always suffice. When the human user is the weakest link, a successful phishing attack may simply need to trick that user to the point that he or she ignores common warning signs (or, in some cases, even the alerts that are shown by an anti-phishing system). Second, due to the wide availability of exploitable Internet infrastructure and lack of universal authentication in e-mail protocols [117] and other messaging platforms [19], phishing attacks can be carried out on a large scale and at a low cost to attackers. Finally, attackers generally have the upper hand as they greatly outnumber defenders, and their mission does not hinge on successfully phishing *every* targeted victim—only enough to make a profit. On the contrary, the organizations that are impersonated by phishers have a much harder job, as they must be able to pinpoint attacks amid a large volume of benign user activity, all while ensuring the

availability of large online systems. Even a small control gap in such a system could mean victory for the attacker. I discuss the challenges in the fight against large-scale phishing, in detail, throughout the rest of my dissertation. Consequently, I develop a series of systems to overcome some of these challenges and to ensure a high baseline level of phishing protection for Internet users.

## 1.3 Contributions of this Dissertation

I have thus far discussed *why* phishing attacks continue to work. However, to be able to improve existing defenses and seal gaps that are being exploited by criminals, we must understand exactly *how* it is that phishing can continue on such a scale. Due to the difficulty of making representative measurements at the ecosystem level, and the rapidly evolving landscape of threats, much prior research has often glanced over the necessary measurements to gain such an understanding [130].

In this dissertation, I present work that sheds new light on phishing attacks, provides ecosystem-level measurements, and ultimately motivates a new generation of anti-phishing defenses that recognize the importance of proactive threat intelligence and collaboration within the ecosystem. Moreover, through vulnerability disclosures and subsequent collaborations, parts of my research have led to security recommendations that have positively impacted the online security of more than one billion mobile browser users [101].

In Chapter 3, I analyze the evasion techniques employed by phishers through the first in-depth study of *.htaccess files* in the context of phishing websites. I also estimate the extent to which compromised infrastructure is used for phishing and propose methodology for fingerprinting attacks based on URL and WHOIS data. I hypothesize that evasion techniques that might otherwise seem trivial are in fact defeating today's primary anti-phishing defenses, and that more sophisticated techniques may be capable of causing a

far greater degree of damage. In addition, I find that a low barrier to entry due to the easy accessibility of *phishing kits* in underground economies is a key factor in the scalability of phishing attacks.

To test the hypothesized effectiveness of evasion techniques against the anti-phishing ecosystem, I propose and develop a framework called *PhishFarm*, which I describe in Chapter 4. PhishFarm enables the creation of highly-automated experiments to test anti-phishing blacklists—and other mitigations—by introducing batches of representative (albeit innocuous) phishing websites into the ecosystem. This approach allows us to obtain ground truth on the deployment time and configuration of each phishing website, and it can be done without involving any human users or phishing victims. Through the deployment of several experiments consisting of thousands of phishing websites, PhishFarm identified actionable shortcomings in the blacklists that protect modern web browsers and resulted in numerous security recommendations (that I disclosed) for the ecosystem as a whole.

In Chapter 5, given the positive impact of the initial set of findings obtained by PhishFarm, I extend the system to perform a long-term study of the anti-phishing ecosystem and develop a new framework, *PhishTime*, for the continuous generation of longitudinal experiments that replicate the evasiveness of real-world phishing websites. I use PhishTime to verify if blacklisting remains consistent over time, measure the implications of deploying phishing websites with SSL certificates, and test combinations of evasion techniques representative of sophisticated phishing attacks in the wild. In addition, I precisely measure the response time (*speed*) of anti-phishing blacklists to determine the window of opportunity that is available to attackers: a reliable indicator of whether the ecosystem's defenses are improving. Findings from the PhishTime study revealed changes to the ecosystem since the original PhishFarm experiments—some of which directly resulted from the aforementioned security recommendations. Both PhishFarm and PhishTime of-

fer re-usable and scalable methodology which can continuously be used by the ecosystem to bolster the understanding of the state of its defenses.

The response time of mitigations such as anti-phishing blacklists is only a small component of the complete timeline of phishing attacks. To be able to fully understand the extent to which these mitigations are deterring attackers—and to gain further insight into characteristics of successful attacks—deeper analysis is necessary. In Chapter 6, I propose methodology to paint an end-to-end aggregate timeline of real-world phishing attacks, from the time of initial deployment, through the points at which victims interact with attack websites, to the time that victim credentials are monetized. In the first study of its kind, I deploy this methodology on a large dataset of network traffic from a major financial services provider and a dataset of user phishing reports from a major e-mail provider. The findings from this study motivate an increased focus on proactive defenses in the long-term fight against phishing, and the same methodology can be used directly as a proactive anti-phishing mitigation.

On the surface, modern phishing attacks are marked by a seemingly unending volume. Throughout the research presented in this dissertation, however, I subject these attacks to close scrutiny, which reveals a high degree of ingenuity collectively applied by attackers. I show that sophisticated phishing websites are far more effective than simple ones: By prioritizing mitigations targeting these websites, the anti-phishing ecosystem can start closing the key gaps that currently enable phishing to remain profitable as a whole.

Although it may not yet be possible to fully defeat phishing, I hope that the techniques presented throughout my dissertation will influence the next generation of anti-phishing systems to better (and continuously) adapt to ever-evolving and evasive phishing attacks, just as attackers have been adapting to the current generation of anti-phishing mitigations through their array of clever tactics.

# Chapter 2

# BACKGROUND

To be able to make a positive impact on the fight against phishing, we must first familiarize ourselves with the nature of phishing attacks, the tools at the disposal of attackers, and the main defenses currently in place. We must then understand how phishers respond to those defenses so that we can identify potential vulnerabilities and think ahead to the next steps that phishers might take. In this chapter, I introduce basic aspects of phishing and provide an overview of the ecosystem that surrounds phishing attacks.

## 2.1 Social Engineering

In the context of information security, *social engineering* refers to the psychological manipulation or deception of a victim, by an attacker, such that the victim performs an action that is beneficial to the attacker [66]. Phishing is a specific type of social engineering through which attackers (known as *phishers*) seek to trick victims into disclosing sensitive information [27]. Phishing has been fueled by the growth of the Internet and commonly occurs online.

Phishing attacks currently target millions of users each year and can cause direct damage to victims through account compromise, financial fraud, and identity theft [81, 125]. These attacks also inflict reputational damage to brands which they impersonate, and collateral damage to the broader ecosystem, which fights to mitigate the threat posed by the attacks. Modern phishing attacks fall into two general categories, both of which can cause substantial damage to their victims: *spearphishing*, which targets specific high-value individuals, groups, or organizations [52, 55], and *traditional* large-scale attacks, which are

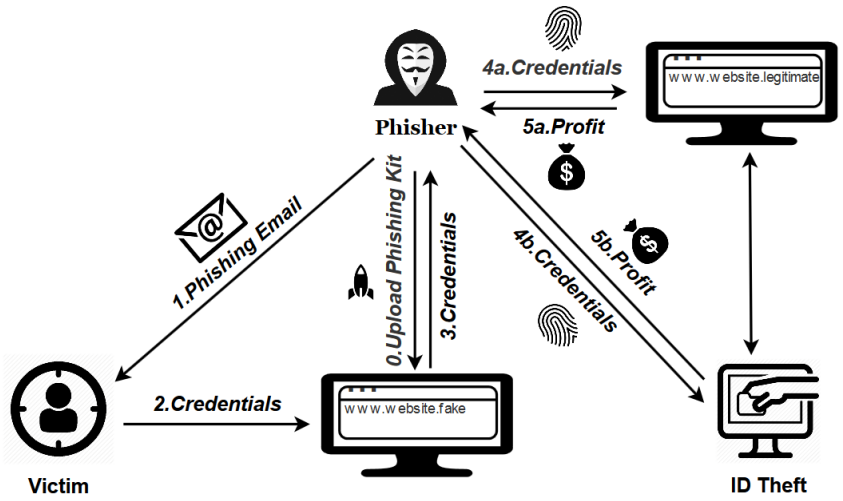



Figure 2.1: A traditional phishing attack.

distributed to a broad range of potential victim users and enable attackers to profit through volume [114]. In this dissertation, I focus primarily on the latter.

## 2.2 Phishing Attack Anatomy

The stages of a typical phishing scenario are illustrated in Figure 2.1. An online phishing attack consists of three main stages: *preparation* (❶), *distribution* (❷ and ❸), and data *exfiltration* (❹).

First, before targeting any potential victims, an attacker spoofs a website by copying its look and feel such that it is difficult for an average user to distinguish between the legitimate website and the fake one (❶). Next, the attacker sends messages (such as spam e-mails) to the user, leverages social engineering to insist that action is needed [27], and lures the user to click on a link to the phishing website (❷). If the victim is successfully fooled, he or she then visits the website and submits sensitive information such as account credentials or credit card numbers (❸). Victims will often be shown a reassuring confirmation message to minimize suspicion of the attack after the fact. Finally, the phishing website transmits the victim's information back to the phisher (❹).

Phishers who obtain victims' credentials will then attempt to fraudulently use them for monetary gain [33] either directly (4a, 5a) or indirectly (4b, 5b) [126].

## 2.3 The Scalability of Phishing

Although phishing attacks might seem dangerous at first glance, the likelihood that a targeted victim will be successfully deceived by phishing is in fact relatively low. In a recent empirical study, only 9% of visitors to a phishing website ended up submitting credentials [51]. The true danger of phishing attacks stems from their *scalability* and a *low barrier to entry*. In other words, it is easy for attackers to target a very large number of potential victims and still make a profit. Such scalability is made possible through underground economies, in which skilled criminals offer commoditized tools and services that enable various types of cybercrime, including phishing [53]. These services enable attackers—even those who otherwise lack technical knowledge—to easily perform phishing attacks and monetize stolen data.

### *2.3.1 Phishing Kits*

A phishing kit is a unified collection of tools used to deploy a phishing website on a web server [82]. Some phishing kits are closely held by their creators, while others are offered as part of the cybercrime-as-a-service economy [74]. Certain criminals specialize in creating and selling phishing kits and will even accept custom requests for kit creation [12]. Kit creators compete based on the effectiveness, ease of use, or perceived security (i.e., from anti-phishing systems) of their kits. Other criminal service providers sell or barter to provide pre-hacked web servers (sometimes called "shells" or "cpanels" in criminal marketplaces). Still others offer lists of spam recipient e-mails and tools for sending phishing messages in bulk [126]. Together, these tools lower the barrier to entry and allow criminals with very minimal technical skills or limited capabilities in English

to become successful phishers [22]. The phisher can simply buy a kit, customize it by replacing the destination e-mail address, and upload and unzip the kit on a pre-hacked web server. The phisher then loads a pre-written message and a list of target e-mails into his or her spamming tool, hits "send," and waits for stolen credentials to arrive in his or her inbox.

Basic components of a phishing kit include a template that mimics the design of the website being impersonated, server-side code to capture and send submitted data to the phisher, and optionally code to filter out unwanted traffic or implement other countermeasures against the anti-phishing community. Such countermeasures might include URL shortening or redirection, URL randomization, human verification, or code obfuscation [51]. I analyze phishing kits in more detail throughout Chapter 3, and I discuss certain sophisticated phishing kits in Section 6.7.1.

### *2.3.2 Deploying a Phishing Attack*

To carry out a traditional phishing attack, phishers first need to have access to a live web server to host the phishing website. In most cases, creating the website merely involves uploading a phishing kit archive to the server and extracting its contents to the desired URL path [40]. Free hosting providers, as well as legitimate infrastructure that has been compromised, are particularly common hosting targets. Such hosting is desirable because using an existing live URL bypasses the requirement to purchase a new domain name and, thus, saves phishers both time and money [89]. Otherwise, the phisher must also register a domain name that will point to the phishing content.

In the case of compromised infrastructure, the phisher gains access to upload malicious files to a web server he or she does not own by exploiting a known web vulnerability or by using default or stolen credentials to access administrative software running on the server [17]. Exploitation is often automated and results in the uploading of a shell script

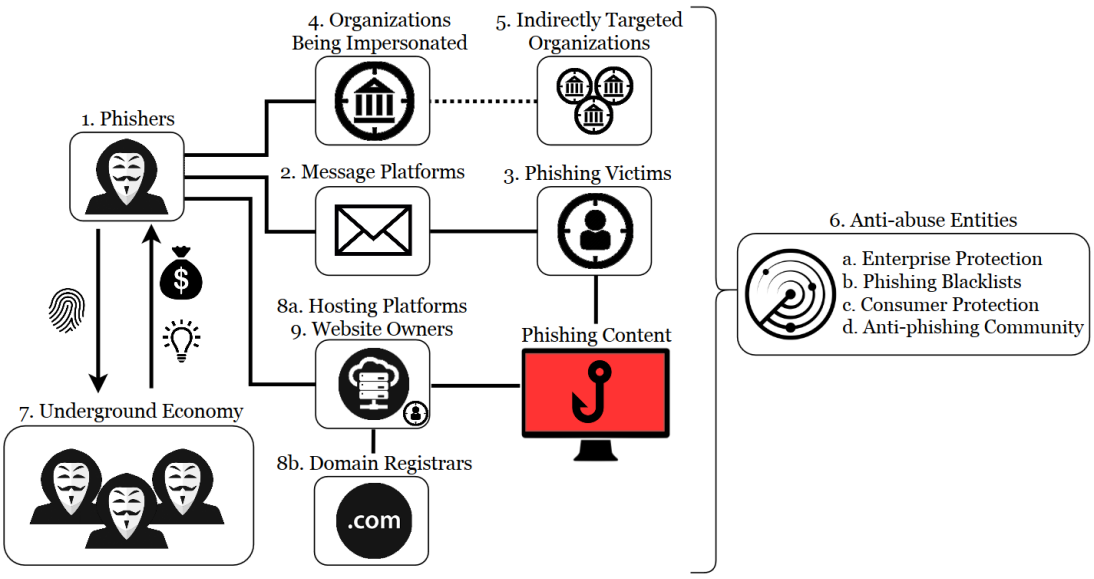



Figure 2.2: Key components of the anti-phishing ecosystem.

on the server which can then be used to remotely execute commands. Service providers in underground economies routinely provide access to such infrastructure for a fee [53].

Once the phishing website is online, phishers distribute its URL through means such as e-mail, social media, or direct messaging [57]. Messages are crafted to deceive the user and often convey a sense of urgency to encourage action [27].

The phishing campaign will remain online for some period of time during which the phishing website collects credentials from victims who fall for the scam and forwards them to the phisher. Eventually, the website will be detected and blacklisted by anti-phishing systems, abandoned by the phisher, or forcefully taken offline by the web host [94]. Security efforts aim to minimize the amount of time that passes between phishing website deployment and blacklisting or take-down [114].

## 2.4 Anti-Phishing Ecosystem and Defenses

Phishers have extensive control over the configuration of the phishing websites that they deploy. As I described in the previous section, this configuration includes the loca-

tion (URL and hosting provider) of the website, the software the website uses to display the malicious content and capture user input, and the deceptive messages distributed to victims. Adversaries, entities who seek to fight phishing, and organizations impersonated or otherwise abused by phishers are involved in various ways in each of these three areas. The relationships between these entities give rise to a complex anti-phishing ecosystem which, due to its scale and potential impact on mitigating phishing, remains the center of discussion throughout Chapters 3-6.

### *2.4.1 Ecosystem Overview*

As shown in Figure 2.2, the phisher (❶), phishing message (❷), victim user (❸), and organization being impersonated (❹) lie at the heart of a phishing attack. Without these basic components, there would exist no basis of trust between the victim user and organization and no means of exploitation by the phisher [27].

Users are prone to re-using the same credentials across different services [125]. This means that the damage from each successful phishing attack can potentially cause a chain reaction spanning multiple organizations. Thus, the organization directly targeted by the phisher (❹) expands to a set of indirectly targeted organizations of interest to the phisher (❺) that use the same authentication scheme (such as username and password). Given the risk of damage that arises as a result, the organizations implement mitigation strategies consisting of their own security teams, third-party anti-phishing vendors, or law enforcement (❻a). Because the victims themselves are the weakest link in any phishing attack, such mitigations are not always technical: they also include user education and phishing awareness training [141].

### 2.4.2 Detecting Phishing Content

Phishing websites are displayed to the user through a web browser, which, in turn, necessitates browser-based defenses. To support these defenses, there exist organizations that maintain blacklists of known phishing websites, organizations that verify phishing reports, and native web browser functionality that checks the blacklists [114] and blocks known websites as a baseline protection against phishing attacks (6b). Consumer-oriented security firms (6c) also offer software for end-users who want additional protection (e.g., antivirus and internet security tools). While the former three classes of organizations all contribute to the anti-phishing effort, they have different priorities and scopes of operation, and are thus worth distinguishing.

Because native browser blacklists (discussed in far greater detail in Chapter 4) accept user phishing reports, a community of savvy web users, industry collaborators, and researchers joins the fight against phishing (6d). These groups gave rise to organized community-driven efforts to list and confirm phishing websites, such as PhishTank, OpenPhish, or the APWG [29].

Phishing content itself often stems from phishing kits, which can be obtained through forums or dark web communities that fuel cybercrime (7). Credentials and personal data stolen through successful attacks are sold by phishers via illicit underground economies [126] which in turn yield tools and motivation for future attacks.

### 2.4.3 Hosting Infrastructure

The hosting platforms (8a) on which phishing websites get deployed fall into two main categories: those controlled directly by the phisher and abused to carry out attacks, and those belonging to legitimate websites that get hijacked by the phisher [51]. In the first scenario, domain registrars and certificate authorities (8b), both paid and free, can

be the subject of further abuse through malicious domain registrations [78, 100]. In the second scenario, the website owner (❾) may suffer collateral damage, such as disruption of regular business operations or loss of productivity as a result of incident response from the hosting provider or one of the security vendors previously discussed (6a). The hosting provider may also make efforts to reduce hijacking through diligent patching or intrusion detection. Through direct abuse reports [44], the hosting provider may also enact content take-down [1] as an anti-phishing mitigation.

### *2.4.4 Message Distribution*

Phishers require a communication channel to initiate their scams against targeted victims. Major e-mail providers and social media networks inevitably capture a large volume of phishing messages (❷) through their platforms [19], thus they are dragged into the ecosystem once they start dedicating resources to protect their users. E-mail providers may check incoming messages, mark them as spam or alert the user if malicious content is found, and forward abuse reports of identified phishing URLs to concerned entities (6a, 6b) [64, 125]. With the rapidly changing state of social media and mobile devices [96], phishers are also keen to bypass the protections available in traditional communication channels such as e-mail.

## 2.5 Summary

Despite appearing simple on the surface, phishing attacks are not only complex, but they are a widespread problem that remains unsolved from a security perspective, despite ever-evolving mitigations within the ecosystem. Phishers gravitate toward whatever tools and methods most efficiently facilitate their attacks on a profitable scale.

In the next chapter, I use phishers' own tools against them: I leverage a large dataset of phishing kits and URLs to understand the precise nature of phishing within the ecosystem

(between 2016 and 2017), with a focus on key tactics that phishers leverage to slip past defenses while effectively deceiving their victims. Consequently, this understanding will reveal the gaps in defenses that allow phishing attacks to remain worthwhile for criminals. In Chapters 4-6, I focus on studying these gaps in more detail, and I propose ways in which they can be measured and addressed.

# Chapter 3

# UNDERSTANDING EVASION TECHNIQUES THROUGH PHISHING KIT ANALYSIS

## 3.1 Introduction

The behemoth scale of credential theft cannot be overstated. Between March 2016 and March 2017, malware, data breaches, and phishing led to 1.9 billion usernames and passwords being offered for sale on black market communities [125]. Although phishing attacks are conceptually simple, they are difficult to effectively counter because phishers and anti-phishing entities are engaged in an endless cat-and-mouse game. The technological tools used by both are ever-evolving in response to the other's actions [71].

Phishing attacks are particularly damaging not only due to their high volume, but because their impact extends beyond the individuals who are directly targeted. The organizations being impersonated in phishing attacks (such as financial institutions or e-mail providers) expend vast resources to minimize their losses and must work together with security firms and researchers to address attackers' increasing level of sophistication. As I discussed in the previous chapter, this has given rise to an anti-phishing ecosystem comprised of diverse entities working toward the goal of mitigating the threat posed by phishing [18].

In this chapter, I leverage two different datasets to expose the specific techniques that phishers employ to avoid detection by the anti-phishing ecosystem while maximizing their return on investment. In other words, I follow the breadcrumbs that phishers leave in the wild to uncover their tactics. Ultimately, my goal is to obtain a clear understanding of the current ecosystem, the motivations of phishers, potential threats to victims, and possible shortcomings of abuse-reporting entities. I propose general solutions to counter

sophisticated phishing attacks and identify new research directions for evaluating and improving phishing countermeasures: these findings set the stage for later chapters of this dissertation.

I first analyze a dataset of over 2300 real-world *phishing kits* (retrieved by Cofense, Inc. between Q1 and Q2 2016) to gain insight into the different server-side approaches that phishers take to evade existing phishing website detection infrastructure, specifically focusing on filtering directives found in *.htaccess* server configuration files. Many of these directives allow us to identify security organizations commonly targeted by phishers. Because phishing kits are ready-to-deploy, reusable packages used to carry out phishing attacks, I am also able to observe patterns in the distribution and adoption of such kits across multiple attacks.

I then use over 170,000 phishing URLs (submitted to the APWG during the first half of 2017) to identify the extent to which compromised infrastructure and domains are used for phishing. Based on this data, I propose an up-to-date phishing URL classification scheme, and I combine URL classification with domain age to fingerprint each phishing attack.

Analyzing the software code within phishing kits and the corresponding attack URLs allows us to not only understand the goals of phishers but also reveals the evasion techniques that they use. In turn, dissecting these techniques allow us to profile the nature of each attack. Previously, no detailed insight into the server-side evasion techniques (which I discuss in Section 3.2) has been published in the context of phishing, yet thoroughly understanding these techniques can help anti-phishing entities identify and mitigate attacks more quickly and more reliably [58].

Phishers often carefully select the URLs that host their attack websites, either in whole or in part. These URLs are frequently crafted to deceive victims [41], but they can also be formulated to instead evade detection. With minimal effort from the phisher, request

filtering and cleverly chosen URLs can dramatically bolster the effectiveness of a phishing attack; I thus focus my analysis in this chapter on these two areas.

My immediate contributions can help organizations involved in the fight against phishing consider the dynamics within the entire anti-phishing ecosystem, understand what influences patterns in phishers' URL and hosting selections, and understand the nature and purpose of server-side filtering that phishers employ. These findings are the precursors to a larger study in Chapter 4, which seeks to measure the true effectiveness of phishers' evasion techniques and is motivated by the potential for a more effective and timely incident response by anti-phishing entities (to ultimately improve the security of potential phishing victims).

## 3.2 Analysis of Server-Side Filtering

Many phishing kits employ *request filtering*, which requires some set of conditions to pass (based on information contained in the HTTP request or server state) before the phishing website is displayed to the client. Filters are of particular interest to researchers as they allow us to gain insight into the organizations that a phishing kit is trying to evade, or the group of users being targeted, thus making it possible to fingerprint the kit. Filtering in phishing kits is analogous to content *cloaking* [58], a technique used by spam websites to serve unwanted content to human visitors while showing seemingly benign content to web crawlers. Within the latter context, the goal of cloaking is to attain high search engine rankings. Phishers use cloaking in a similar manner, but for a different purpose.

Denying requests through server-side filtering seems paradoxical at first glance, as the phisher's goal is to scam as many victims as possible. However, the phisher also wishes to evade detection, which is where filtering plays an important role. It is thus in the phisher's interest to serve the phishing content to a legitimate victim while denying access

to search engines, security firms, researchers, and blacklist crawlers, all of which could lead to the detection of the attack and trigger an anti-phishing response. Successfully blocking these entities decreases the likelihood of timely detection and blacklisting of the phishing website, ultimately increasing the phisher's return on investment.

Filtering can be implemented in various places, including server directives, server-side scripts (written in languages such as PHP and Python), or JavaScript that runs in the user's web browser. The former two approaches are common and allow for very similar types of filtering with certain trade-offs as discussed below. The latter is an emerging pattern seen in more sophisticated phishing kits and would be suitable for detailed analysis—possibly through scalable automated crawling of phishing website source code—in future work.

*.htaccess files* are used to supply configuration information for the Apache web server, the most common server software used for approximately 44.99% of active websites as of August 2017 [96]. Such files are placed in directories containing web content and scripts. They allow configuration to be specified without root-level access to the server, which makes them possible to deploy without a complete breach of a system (gaining access to upload files to a public folder is generally sufficient). They are also simple for phishers to write and maintain through different iterations of a kit as they carry no dependencies. All this ease makes .htaccess files particularly appealing and they are therefore a common sight in phishing kits. .htaccess files lend themselves well to analysis as they are comprised nearly entirely of filtering directives in a homogeneous format, as opposed to arbitrary server-side code which can be written in many different ways and can be obfuscated.

I examined a dataset of PHP scripts from phishing kits as part of this research and found them to implement filtering strategies in the same manner as .htaccess files. The main benefit of using scripts over server-side directives is the ability to track a user, something which has been previously studied [22, 51]. I thus focus my analysis on .htaccess files.

In the following sections, I examine a large dataset of .htaccess files in detail to reveal the nature and prevalence of request filtering techniques employed by phishers while identifying their underlying motivation. Consequently, I propose methods to defeat each type of filter identified, and I synthesize a list of the anti-phishing organizations that phishers attempt to evade based on the contents of the .htaccess files. Finally, I consider metadata of the .htaccess files to plot phishing kit age and evaluate kit re-use.

### *3.2.1 Dataset Overview*

My dataset consists of a sample of 2,313 .htaccess files extracted from 1,794 live phishing kits hosted on 933 different domains. These kits were retrieved between January 1 and June 30, 2016, and provided to me by Cofense (formerly PhishMe), a company that specializes in phishing-related security solutions. In addition to the contents of the .htaccess files, the dataset contained the file modification date, date of retrieval, and URL of each kit.

```
1  order allow,deny
2  allow from all
3  deny from 60.51.63.      # websense bandwidth waster
4  deny from 87.233.31.45   # bot rips way too fast
5  deny from 46.134.202.86
6
7  deny from paypal.com
8  deny from apple.com
9
10 RewriteEngine on
11
12 RewriteCond %{HTTP_REFERER} google\.com [NC,OR]
13 RewriteCond %{HTTP_REFERER} firefox\.com
14 RewriteRule .* - [F]
15
16 RewriteCond %{HTTP_USER_AGENT} ^googlebot
17 RewriteRule ^.* - [F,L]
```

Listing 3.1: Partial .htaccess file with all four types of blacklist filters and real comments left by phishers.

| Approach | Filter Type | All Files (2313) | | Unique Files (153) | |
|---|---|---|---|---|---|
| | | **Filter Count** | **Files w/ Filter** | **Filter Count** | **Files w/ Filter** |
| Blacklist | Deny IP | 1,046,397 | 2194 | 234,125 | 98 |
| Blacklist | Hostname | 16,540 | 913 | 794 | 76 |
| Blacklist | Referrer | 4,177 | 572 | 976 | 48 |
| Blacklist | User Agent | 111,255 | 462 | 11,904 | 41 |
| Whitelist | Allow IP | 315,907 | 9 | 94,515 | 5 |

Table 3.1: Occurrence of different filter types in the .htaccess files.

### *3.2.2 Cleaning the Data*

Because .htaccess files consist of plain text and generally contain one directive per line, they are straightforward to parse. I started by identifying duplicate files in the dataset after stripping comments and empty lines (I preserved inline comments as metadata for later analysis). This left 153 (6.6%) unique .htaccess files out of the total of 2,313. Of these unique files, 73 were seen only once, 66 appeared an average of 7.5 times, and 14 outliers appeared an average of 124 times. The outliers were attributable to a handful of kits with a large number of sub-directories that all contained the same .htaccess file.

I then identified syntactic variations of directives with the same semantics (such as IP block rules) and iterated through each .htaccess file to obtain an aggregate overview of its filters.

### *3.2.3 Filter Types and Frequency*

I discovered five different major types of filters used in the .htaccess files, distributed as shown in Table 3.1. The most common *deny IP* filter takes a blacklist approach to block requests from specific IP addresses, partial IP addresses, or CIDR ranges [108]; at least one such rule was present in 64% of the unique files and 95% of all files. Three other blacklist filters checked the client's *hostname*, *referring URL*, or *user agent string* to deny requests

matching the specified strings. On the opposite end of the spectrum, the *allow IP* filter took a whitelist approach to grant access only to specific IP addresses. In the case of my dataset, the *allow IP* filter was only present in a handful of files that performed geolocation to restrict traffic to single countries. The four blacklist filters provide insight into the specific entities that phishers are trying to exclude. On the other hand, the whitelist filters instead reveal the location of the victims of the phishing scam.

In addition to these filters, I found that 30% of the unique .htaccess files, and 61% of all files, limited HTTP request methods to *GET* and *POST*. For the sake of completeness, other less interesting directives included the specification of an index script, disabling of server-level directory indexes, and allowing only requests for certain file extensions. The disabling of indexes can serve as a deterrent to the scanning of phishing websites (and, potentially, victim data or phishing kit archives) by anti-phishing crawlers.

### Organizations of Interest

In this section, I discuss the nature of the blocked IP addresses, hostnames, referring URLs, and user agents as I unmask why phishers selected them as part of their filtering strategy.

### IP Address

I identify the entity targeted by each IP address through a variety of techniques: performing a reverse DNS lookup to obtain the hostname, visiting the IP, querying an ISP or IP geolocation database (I used IP2Location and GeoLite2, respectively [59, 79]), and manually interpreting the comments left behind by phishers. For some 4,300 IP addresses out of the total of 29,971 unique blacklisted IPs I extracted, phishers included a comment describing the entity believed to be tied to the address. These comments could be found in 23% of the files in the dataset. Examples of IP filtering and the corresponding comments are found in Listing 3.1.

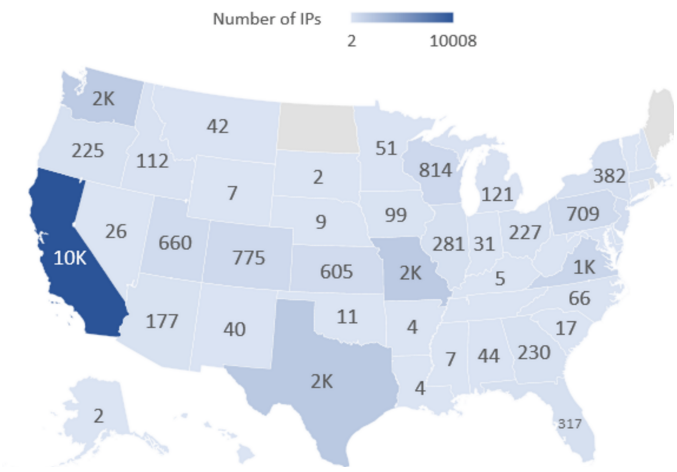



Figure 3.1: Distribution of blocked IPs in the United States.

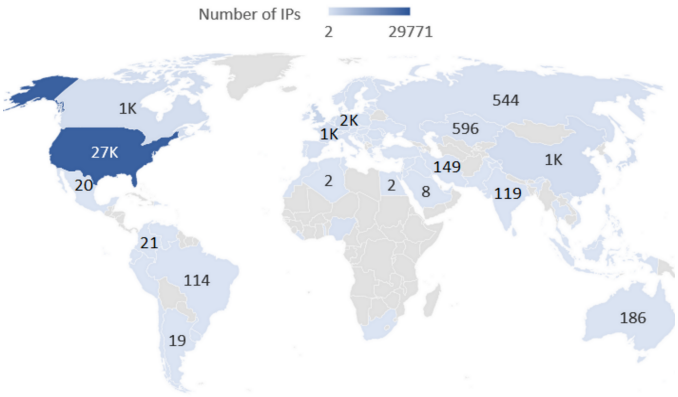



Figure 3.2: Distribution of blocked IPs worldwide.

In my analysis, I combined all of these techniques to obtain as much information as possible about each IP address. For each IP address in the dataset, I recorded the frequency as well as the associated entity. I then categorized these entities based on their primary business type.

Analysis of the data revealed that the phishers developing these .htaccess files focus heavily on blocking requests from web hosts, web crawlers, and internet service providers, with a secondary focus on security companies, universities, and organizations involved in DNS administration. Per the GeoLite2 database, over 90% of the unique IP addresses

in the dataset were located in the US, with approximately half originating in tech-heavy California, as shown in the maps in Figures 3.1 and 3.2. Areas with concentrations of blocked IPs coincide with the headquarters or data centers of major organizations involved in internet security.

Avoiding traffic from these entities is important to phishers because detection by a web host might lead to deactivation of the platform hosting the phishing website or identification of the person behind the attack [94]. Detection by a search engine crawler or security company would likely result in blacklisting of the phishing content, which could terminate the phishing campaign before the phisher finishes his or her work [51]. Generally speaking, the phisher does not want anyone but the victims to be able to access the phishing page.

I can conclude that phishers make a considerable effort to identify and attempt to bypass the anti-phishing infrastructure being used against them. While this dataset does not allow us to make any measurements of the effectiveness of phishers' evasion efforts, the security industry can regardless respond by using a diverse and ever-changing network of systems and IP addresses.

### Hostname

By total count, filtering by hostname (of the IP address of the user making the HTTP request) was the least common filter type in the dataset. This rarity is likely because such filters require the server to perform a reverse DNS lookup for every HTTP request which is costly in terms of time and could impact the availability of the phishing website. However, nearly half of the unique .htaccess files contained at least one hostname filter, suggesting that phishers trust their effectiveness.

Hostname filters showed a heavy bias toward the victim organizations as well as anti-phishing organizations, and also included some antivirus vendors. Some were designed

to match keywords that might be present in a security-related hostname, such as "phish," "spam," or ".edu."

This filter can be evaded by ensuring that no *PTR* reverse DNS record is configured for the IP address accessing the kit, or that the record is not revealing.

### Referrer

When a human makes a web request by following a link in a browser, the browser will typically transmit the URL of the referring web page in the *Referer* HTTP header [40] of the new request. For instance, if an employee at a security company were to manually verify a phishing URL by clicking on a link in an e-mail or internal database, the company's name may be revealed in the referring URL. Phishers can take advantage of this behavior by blocking requests containing certain referring URLs, as shown in Lines 12–14 of Listing 3.1.

I found that the referrer filters focused exclusively on antivirus companies, security companies, and blacklist providers. These filters merely contained the primary public domain of these companies (e.g., *google.com* or *mcafee.com*), which suggests that phishers may have been guessing rather than basing these filters on known referrers.

The anti-phishing industry can easily bypass such filtering by configuring browsers or crawlers used for phishing detection to never transmit referrer information or to transmit a benign-looking URL.

### User Agent

The user agent string (defined in RFC 2616) identifies the software issuing the HTTP request on behalf of the user, such as a browser or robot [40]. In the dataset, user agent filters were used exclusively to block known web crawling and scraping software.

Table 3.2: The most commonly observed entities in the .htaccess dataset.

| Frequency | Ecosystem Entity | Type |
|---|---|---|
| 257 | Google | Crawler<br>Blacklist |
| 96 | PayPal | Victim Org. |
| 91 | Internet Identity | Security |
| 81 | Bit Defender | Security |
| 49 | McAfee | Antivirus |
| 42 | Forcepoint | Security |
| 42 | Mark Monitor | Security |
| 39 | Brand Protect | Security |
| 37 | Looking Glass Cyber | Security |
| 35 | AVG | Antivirus |
| 34 | Eset | Antivirus |
| 33 | Kaspersky | Antivirus |
| 27 | Firefox / Mozilla | Browser |
| 25 | TrendMicro | Antivirus |
| 22 | Apple | Victim Org. |
| 21 | Symantec | Antivirus |
| 21 | Netcraft | Security |
| 20 | F-secure | Antivirus |
| 19 | Dr. Web | Antivirus |
| 15 | Avast | Antivirus |
| 14 | Avira | Antivirus |
| 14 | ClamAV | Antivirus |
| 12 | Spamcop | Security |
| 11 | Yandex | Crawler |
| 11 | Comodo | Security |
| 10 | Microsoft | Blacklist |
| 10 | PhishTank | Security |

Lines similar to 16–17 in Listing 3.1 were commonly found in the dataset and seek to block the Google crawler. I identified no other references to the anti-phishing entities that I saw in prior filters. This absence suggests that although phishers certainly wish to prevent automated tools (such as *wget* or a software library) from fetching their websites, they perhaps do not know the specific user-agent strings used by security infrastructure, or they know how trivially these can be changed. Regardless, because the user agent string can be spoofed, phishing blacklist crawlers can and should frequently take advantage of varying the user agent in their requests.

### *3.2.4 Combined Summary*

In Table 3.2, I summarize the most frequently observed entities in the dataset of 153 .htaccess files, specifically those with ten more distinct appearances. I aggregated the data by assigning a unique identifier to each entity, then combined the total number of occurrences of each entity within each filter type. Many of the entities listed are of integral importance in the anti-phishing landscape. Of particular note are Google Safe Browsing and Microsoft SmartScreen, which operate the blacklists that natively protect Google Chrome, Safari, Firefox, Internet Explorer, and Edge, and account for over 97% of global desktop traffic as of August 2017 [119]. Similarly, it is no surprise that PayPal and Apple appeared in this list as these companies ranked second and third as the most targeted brands in the APWG dataset, respectively, and were common victims in prior studies of phishing kits by Cova et al. [22] and Han et al. [51]. The large, user-driven anti-phishing community PhishTank saw a disproportionately low representation in .htaccess files, possibly due to the distributed nature of the community [29].

It is evident that phishers work hard to thwart anti-phishing efforts by evading detection to ultimately bypass the defenses offered by blacklists, security firms, and antivirus vendors. Furthermore, it is arguably eye-opening that such a clear picture can be painted

from data created by and known to phishers. At the same time, the .htaccess data does show some cracks: filters in older kits (discussed in the following section) lag behind in the past, as many references are made to defunct companies or companies that have merged with others within the past two years. Measuring the true effectiveness of phishers' filtering techniques within the ecosystem is an interesting research problem; I propose methodology for doing so in Chapter 4.

### *3.2.5 Phishing Kit Sharing and Age*

I found that, on average, 87 days elapsed between the first and the last retrieval of the duplicate .htaccess files discussed in Section 3.2.2. Furthermore, almost every kit was retrieved from a distinct domain. I can thus conclude that phishing kits see regular re-use, potentially as part of a single phisher's campaign.

Frequent phishing kit re-use is further supported by analyzing the modification date metadata of the .htaccess files. The majority of files were last modified over a year before deployment, as shown in Figure 3.3. Interestingly, a handful of kits dated as far back as 2009. Regular changes to the configuration of anti-phishing infrastructure would quickly render the filtering efforts in such kits obsolete.

## 3.3 Analysis of Phishing URLs

Cleverly chosen URLs can be used by phishers to deceive victims or to make a phishing page appear benign in the absence of other contextual data. In this section, I propose an up-to-date classification scheme that reflects the latest trends in phishing URLs, as motivated by evolving social engineering techniques that are aided by specific URL contents. Moreover, I show that analyzing the domain age alongside the classification of these URLs can reveal information about the infrastructure being used to host phishing content. Patterns in phishing URLs have been shown to allow for automated classifica-

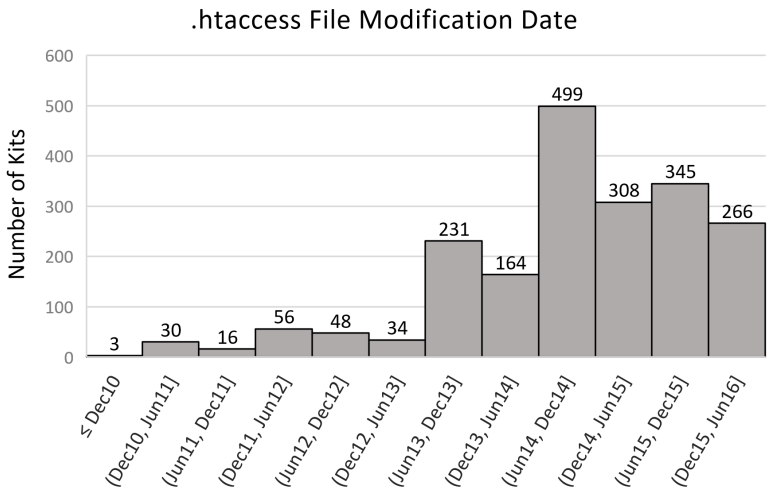



Figure 3.3: Histogram of .htaccess file modification dates.

tion and detection of phishing content, as studied in the work discussed in Section 3.4, and such classification can be enhanced by considering recent developments in URL crafting and understanding the intent behind each type of phishing URL.

### *3.3.1 Dataset Overview*

The APWG has been monitoring trends in phishing and other cybercrime since 2003. After the explosion in the volume of phishing attacks in the early- to mid-2010s, the APWG launched its online eCrime Exchange database, which is a platform—available to APWG members—through which organizations engaged in anti-phishing can share detections of Phishing URLs. As of mid-2017, this database contained over 2.7 million real-world phishing attacks reported since 2015. Each entry consists of a phishing URL, the organization targeted, the date the attack was detected, and a confidence score. I focus on 172,620 URLs submitted during the first six months of 2017, which I fetched from the database on a daily basis and annotated with the domain registration date based on WHOIS data [78]. I further narrow my focus to traditional phishing URLs (i.e., those hosted on web servers)

| | |
|---|---|
| **Type I** | *IP address as hostname, deceptive path contents* |
| | (a) http://66.196.233.2/www.paypal.com/webscr.html?cmd=_login-run |
| | (b) http://93.182.172.145/info/Verify.php?=&securessl=true |
| **Type II** | *Random domain, deceptive path contents* |
| | (a) http://resqplus.net/css/ssl/secure.paypal.com.au/au/cgi-bin/webscr/ |
| | (b) http://www.nae4ha2012.com/logos/login/secure.my.private-settings.support |
| **Type III** | *Long, deceptive subdomain* |
| | (a) http://statements.visa.com.upetkiti.be/cards/myvisa/transactionsphp |
| | (b) http://https.secure.update.customer-update.extrasecure.profilecontinue.charityliberia.org/ |
| **Type IV** | *Deceptive domain name* |
| | (a) http://change-paypal.com/ma/webapps/mpp/home/ |
| | (b) http://support-center-confirm.com/support/Payment-update-0.html |
| **Type V** | *Unintelligible URL* |
| | http://offto.net/5d8ucl/?action=redirect&nick=5d8ucl&m=1 |
| | http://69.93.204.33/~startrac/cash69.html |

Table 3.3: High-level classification scheme for phishing URLs, with examples from the APWG dataset.

rather than social media URLs or redirection links, which are also recorded by the APWG but may not be directly indicative of individual phishing websites [19].

A key aspect of this dataset is that each URL is paired with the targeted organization, which possible because these URLs are primarily submitted to the database by agents of the organizations themselves. These agents may also use classification techniques to confirm the presence of brand names or logos on the corresponding phishing websites. The pairing of URLs with brands allows us to reliably identify the presence of the brand within each URL's hostname or path, and, thus, I address a limitation of phishing URL datasets used in prior work [22, 114].

After removing partial URLs without a hostname, social media URLs, and entries with a confidence score below 100%, I parsed each URL to generate additional attributes: the path, hostname, domain name, top-level domain (TLD), and subdomain level [87]. I then removed hostnames as duplicates if they differed only in the presence of a hash or user ID in the subdomain, a common technique employed by phishers to evade blacklist hits [51]. This pruning left 66,752 unique hostnames. Finally, I used substring extraction [109] to identify common tokens in the URLs and manually classified them as *brand* if they appeared similar to the targeted brand name or *misleading* if their meaning was related to online account security and could potentially be misleading to a user (e.g., "secure," "https," "service", "support"). After classifying all the individual tokens, I created two binary parameters to indicate their presence in the path and hostname, respectively, of each URL. With these parameters, I could then semantically classify the nature of each URL as a whole.

### *3.3.2 Classifying Phishing URLs*

Using the attributes I added to the URL dataset, I propose a phishing URL classification in Table 3.3 that builds on the model of Garera et al. [41], an early and now outdated classification that identified four different types of hostnames in a phishing URL.

In an effort to comprehensively capture recent and emerging patterns in phishing URLs, I identify URLs as one of five mutually exclusive semantic types by looking at both the hostname and path. Types I through IV are divided into a further two sub-types: the more common (a) for URLs recognizably containing the brand name, or (b) URLs containing misleading keywords. Both sub-types aim to trick a user to visit the URL. It was common for the (a) sub-type to contain a slight misspelling of the brand such as "paypol" instead of PayPal or "appel" instead of Apple. In this manner, attackers can directly register deceptive domain names and, additionally, perform *typosquatting* [122]. Such do-

mains may also evade heuristic classifiers that expect the exact brand name [64] or specific keywords [138].

Apart from the introduction of the two sub-types for each, Type I, II, and III URLs are otherwise unchanged compared to Garera et al.'s classification [41]. Type IV URLs contain a deceptive domain name registered by the phisher. Type V URLs are unintelligible in the absence of other metadata and contain a seemingly random hostname (which can be either a domain or IP address) and no brand or deceptive keywords.

Most browsers show the URL of the current page to the user, but the way the URL is displayed differs across browsers. Phishers may thus opt for a specific URL type depending on the targeted browser or platform. For example, the Google Chrome desktop browser highlights the domain name, so a Type IV URL might appear legitimate to unsuspecting users. A Type III URL would be suitable for a mobile browser that only shows the first several characters of the URL due to screen width limitations [131]. For instance, a domain name of *fakesite.com* would not be visible in the displayed portion of a URL such as *www.paypal.com.signin.fakesite.com* when viewed on a small screen. A Type I, II, or III URL would be appropriate for display in e-mails, due to the possibility of long deceptive strings spanning much of the URL. Finally, a Type V URL can be advantageous for evading detection tools that check for specific patterns. Thus, I have identified not only technical reasons for URL selection, but also motivations deeply rooted in social engineering.

In the entire dataset, I identified 156 Type I, 6,899 Type II, 4,186 Type III, 14,289 Type IV, and 41,213 Type V URLs. However, these numbers are far more meaningful when viewed alongside the age of the domain name within the latter four URL types, as discussed in the following section.

### 3.3.3 New vs. Compromised Infrastructure

Although URL content and domain age have historically been used in phishing website classifiers based on machine learning, other heuristic attributes based on web page content and search engine metadata have proven to be much stronger indicators of a phishing attack [138]. Therefore, rather than using the URL type and domain solely to identify phishing attacks, I propose a different use for these attributes: inferring the underlying hosting infrastructure, which can then be leveraged to mount an appropriate incident response by anti-phishing entities (after a URL has been confirmed to contain phishing).

I found that 28.9% of the URLs in the dataset were reported within 1 month (30 days) of the domain registration, while another 53.3% of URLs had domains older than a year. The remainder was fairly uniformly split between 1 and 12 months. These findings are consistent with Hao et al.'s analysis of spam URLs distributed via e-mail [53] and show that attackers' use of compromised infrastructure remains a significant problem. Although it is tempting to outright conclude that old domains belong to benign websites that have been compromised, a much stronger case can be made if I also consider URL classification along with the requirements to deploy each URL type.

In Figure 3.4, I plot the domain age versus the URL type in the APWG dataset. I find that Type IV (deceptive domain) URLs are the most common in newly-registered phishing domains, and that Type III (long subdomain) URLs also occur more frequently with new domains. As domains age, the frequency of Type II (deceptive path) and Type V (unintelligible) URLs increases significantly while other types decline. Type I URLs are omitted from the figure as IPs do not have registration dates, unlike domains; they were also rare in the dataset as a whole (0.23% of unique URLs).

Type I, II, and V URLs only rely on the folder structure of uploaded files, which can easily be controlled directly by phishing kits. Exploiting a web vulnerability to upload a

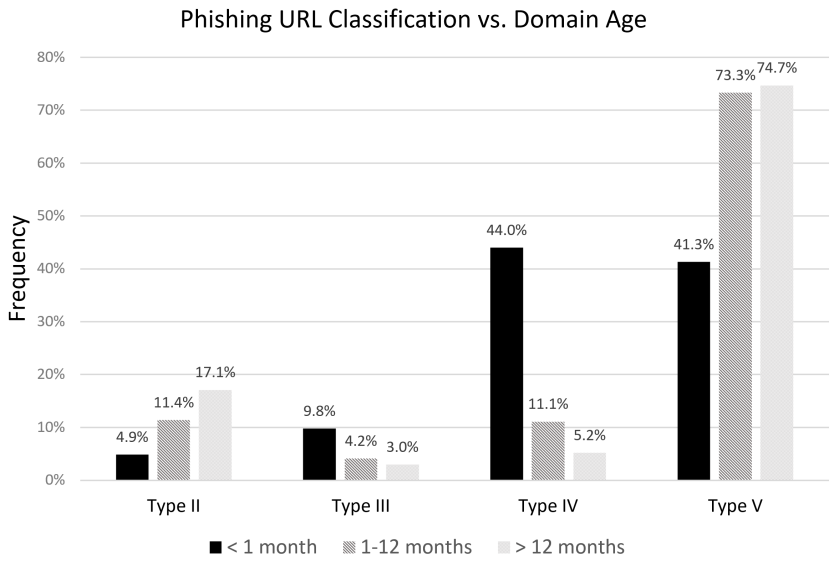



Figure 3.4: Impact of domain age on the distribution of URL types (Type II: random domain with a deceptive path; Type III: deceptive subdomain; Type IV: deceptive domain name; Type V: unintelligible URL).

phishing kit would grant the access necessary to deploy such URLs. On the other hand, a typical web hosting environment requires DNS changes for configuring Type III and Type IV domains; such access would necessitate a larger-scale breach for deployment on a compromised system. I, therefore, hypothesize that older Type II and Type V phishing URLs generally represent compromised infrastructure, while newer Type III and Type IV URLs are more likely to be found on infrastructure deployed by phishers. Although I do not verify this correlation, a future study involving a dataset with attributes such as the content and search engine rankings of each URL could be used to definitively classify the underlying infrastructure.

An intriguing case is my observation of a small proportion of Type IV URLs with domains older than 1 year. Because paid domain names must be renewed annually, it would not make economic sense for a phisher to pay renewal fees prior to using the domain name for the first time. After analyzing the TLD distribution of these URLs, I found that nearly

all of these domains had free ccTLDs including *.tk*, *.ga*, *.ml*, *.cf*, and *.gq*. Such domains are owned by the registrar rather than the phisher and can often be renewed for free or re-acquired for free following expiration; a clear secondary advantage is their anonymity, because no payment details need to be provided by attackers. A handful of outliers within these URLs included legitimate websites that happened to contain a commonly-phished brand name, such as *apple-medical.com*[1]. Type IV URLs consisted of 41.8% *.com* domains (lower than the 47.0% for the entire dataset) and 14.6% of the aforementioned ccTLDs (higher than the 5.1% dataset average). It is also worth noting that these ccTLDs are administered by only a handful of commonly-abused registrars, which may aid in the mitigation of any corresponding domains used by phishers.

In this dataset, *.coms* dwarf the second most common TLD, *.net*, which accounted for only 3.6% of URLs. Full TLD statistics for this dataset are publicly available from the APWG [4].

## 3.4 Related Work

The work most closely related to my analysis is that of Thomas et al. [125], who carried out a comprehensive study of the underground credential re-use ecosystem. As part of their study, the authors briefly discuss web cloaking through IP address blacklists in .htaccess files found in phishing kits, but they do not delve into specific details. Cova et al. [22] performed an analysis of phishing kits freely available through underground sources or left behind by phishers as archives on live websites. The authors focused on identifying backdoors in these free phishing kits and found trends in e-mail provider usage, victimization, drop technique, and URL type. While they considered URL obfuscation techniques implemented through PHP code as a blacklist deterrent, they did not analyze

[1]Such domains could represent false-positive phishing detections, or may have been specifically targeted by attackers for subsequent use in phishing.

if the kits performed request filtering. Han et al. [51] collected a large dataset of phishing kits through a honeypot server to study the anatomy and timeline of phishing attacks in great detail, which is important in understanding how to best respond to a phishing attack.

Much prior research has studied URLs used for phishing attacks. Hao et al. [53] analyze the domain registration behavior of real phishers and reveal commonly abused registrars. Garera et al. [41] provide a high-level classification scheme for phishing URLs. Further studies developed effective machine learning techniques to perform the classification itself [13, 38, 138] in order to automatically identify malicious websites on a large scale. Such systems are among those that respond to phishing content reported to the security community [136]. With the boom of social media in recent years, shortened redirection URLs and social media links have also seen prominent use in phishing, as studied by Chhabra et al. [19].

To the best of my knowledge, no prior work has analyzed *.htaccess files* in detail in the context of phishing. With respect to phishing URLs, I expand on the classification of Garera et al. [41] by introducing a new URL type frequently observed in the APWG dataset, and I combine methodology originally proposed by Matsuoka et al. [78] to further identify the URLs deployed on compromised infrastructure. My datasets are unique because they are based entirely on real-world, live phishing attacks and contain metadata which has been carefully curated.

## 3.5 Discussion

Server-side request filtering can easily be deployed through .htaccess files placed inside distributable phishing kits, which spread quickly due to their appeal to phishers with limited programming knowledge. Fortunately, countermeasures exist for each filtering technique, as discussed in Section 3.2.

As web browsers and anti-phishing technologies continue to mature, phishers respond by adapting their URLs for maximum effectiveness against their victims while simultaneously aiming to avoid detection. Automated phishing website classification systems should likewise adapt to detect more subtle phishing URLs with partial brand names, homographs, and deceptive keywords. The generic URL classification scheme that I propose addresses common modern-day social engineering techniques and can be used as a part of such automated systems. Specifically, while modern web browsers leverage native anti-phishing blacklists to protect users from known malicious URLs, classification based on the URL alone is not widely used [138]. My findings could be used to reduce the potential false-positive rates of such classification by considering attributes that might explain the use of a specific URL type, such as the device or browser used in the request.

Continuous monitoring of recent phishing URLs can further refine my URL classification scheme over time by identifying changing trends. With the evolution of the use of social media platforms and URL shortener services in phishing [19], future analysis should also focus on how these new platforms are used alongside traditionally-hosted phishing websites (I perform preliminary work in this direction in Chapter 6).

The URL-based attributes that I have identified could be incorporated alongside existing detection techniques such as passive DNS [135] in order to respond during earlier stages of a phishing campaign, or with greater confidence. Also, considering the age of the domain along with the URL's classification can help determine whether a legitimate website has been compromised and would allow for an appropriate response by the hosting provider. Finally, the ecosystem could leverage URL-based predictions about the intended phishing victims in an attempt to bypass some of the server-side filtering techniques discussed in Section 3.2.

Phishers rely heavily on compromised infrastructure to carry out their work, but they also obtain their own domains through key bottlenecks including *.com* registrars and free

domain providers. Further efforts should be focused on quickly detecting and responding to malicious registrations, especially when no payment is required. With access to their own domains, phishers otherwise have full flexibility when it comes to crafting a deceptive URL.

## 3.6 Limitations

My findings should be considered with certain limitations in mind. The APWG database, while large and maintained by security professionals, is skewed toward organizations [4] that directly or indirectly partner with the APWG. It contains malicious URLs reported by the community, which are a subset of all phishing attacks, and may be skewed toward URLs detectable based on the methodology used by this community. Also, we must trust the confidence levels assigned by APWG submitters as an indication that the phishing URL listed was a true positive, because in many cases, the phishing content has already been removed by the time a URL appears in the database, so it cannot be re-verified.

The Cofense dataset is a year older than the APWG dataset. However, Cofense uses a brand-agnostic approach to verifying and documenting phishing attacks which is divorced from its customer list. While many or even most phishing websites may contain .htaccess files, once deployed, a .htaccess file is not retrievable via a web request. Thus, I have only analyzed those files which were found in phishing kit archives that could be retrieved from live phishing websites (i.e., those found zip files that a phisher uploaded to a web server, and subsequently extracted). Consequently, it is also possible that the .htaccess files ultimately deployed on phishing websites could differ from those retrieved by Cofense, though this is limitation partially offset by the sheer size and diversity of the dataset.

Despite their potential shortcomings, datasets such as those I considered in this chapter are highly relevant to the landscape of modern phishing attacks. These datasets cor-

respond to the organizations most commonly targeted by phishers and therefore warrant future analysis by the research community. Furthermore, given my finding that phishers' filtering techniques target the most prominent entities involved in the fight against phishing, future datasets built collaboratively from multiple sources could prove to be even more eye-opening.

## 3.7 Summary

In this chapter, I have studied server-side request filtering techniques employed in phishing kits, which has painted a picture of the anti-phishing ecosystem from the perspective of criminals. I observed that the ecosystem spreads far beyond just the victims and organizations that phishers target, yet these phishers have a keen awareness of the tools being used against them by the security community. Phishers seek to maximize their return-on-investment by avoiding detection by tools they know of, increasing the volume of attacks by using compromised infrastructure when possible, and bolstering the effectiveness of attacks by registering highly-deceptive domain names (preferably at no cost) to trick their victims. Security researchers and all involved entities must understand phishers' tactics to be able to respond appropriately to thwart them before phishing methodology evolves further.

It is clear that phishers have a wide gamut of paths available to them when it comes to deploying their attacks. My study provides the building blocks for an enhanced, custom-tailored response to phishing attacks that can be aided by automated technologies. Identifying a URL as malicious is only a basic first step. By combining my URL classification scheme and analyzing domain age, we can profile not only *where* a phishing attack likely originated in terms of infrastructure, but also *why* that URL was chosen. Building on my .htaccess findings, a crawler that is able to bypass and profile server-side filtering efforts can then reveal information about *who was targeted* for a faster detection response time.

Knowing this information will allow anti-phishing efforts to focus on the most effective response, potentially saving the ecosystem time and money while improving the security of the end-user.

In the next chapter, with motivation from the findings in my analysis of .htaccess evasion techniques, I measure how effectively key security organizations in the anti-phishing ecosystem (from Table 3.2) implement countermeasures to phishers' attempts at evasion, with a focus on the real-world security of the end-user. If anti-phishing tools do not effectively bypass (i.e., defeat) request filtering, attacks may not be detected or stopped in a timely manner. In turn, phishers would be able to profit by wreaking havoc upon their victims.

Chapter 4

# PHISHFARM: A FRAMEWORK FOR MEASURING EVASION OF ANTI-PHISHING BLACKLISTS

## 4.1 Introduction

Today's major web browsers, both on desktop and mobile platforms, natively incorporate anti-phishing blacklists and display prominent warnings when a user attempts to visit a known malicious website. Due to their ubiquity, blacklists are a user's main (and, at times, *only*) technical line of defense against phishing. Unfortunately, blacklists suffer from a key weakness: they are inherently reactive [114]. Thus, a malicious website will generally not be blocked until its nature is verified by the blacklist operator. As we observed in the last chapter, phishing websites actively exploit this weakness by leveraging evasion techniques known as *cloaking* [58] to avoid or delay detection by blacklist crawlers. Cloaking has only recently been scrutinized in the context of phishing [102]; previously, there have been no formal studies of the impact of cloaking on blacklisting effectiveness (despite numerous empirical analyses of blacklists in general). This shortcoming is important to address, as cybercriminals could potentially be causing ongoing damage without the ecosystem's knowledge.

In this chapter, I carry out a carefully controlled experiment to evaluate how ten different anti-phishing entities respond to reports of phishing websites that employ cloaking techniques representative of real-world attacks. I measure how this cloaking impacts the effectiveness (i.e., website coverage and speed) of native blacklisting across major desktop and mobile browsers. I performed preliminary tests in mid-2017, disclosed my findings to key entities (including *Google Safe Browsing*, *Microsoft*, browser vendors, and the *APWG*),

and conducted a full-scale retest in mid-2018. Uniquely and unlike prior work, I created my own (innocuous) PayPal-branded phishing websites (with permission) to minimize confounding effects and allow for an unprecedented degree of control.

My study revealed several shortcomings within the anti-phishing ecosystem and underscores the importance of robust, ever-evolving anti-phishing defenses with good data sharing. Through my experiments, I found that cloaking can prevent browser blacklists from adequately protecting users by *significantly decreasing the likelihood that a phishing website will be blacklisted*, or *substantially delaying blacklisting*, in particular when geolocation- or device-based request filtering techniques are applied. Moreover, I identified a gaping hole in the protection of top mobile web browsers: shockingly, mobile Chrome, Safari, and Firefox failed to show *any* blacklist warnings between mid-2017 and late 2018 despite the presence of security settings that implied blacklist protection. As a result of my disclosures, users of the aforementioned mobile browsers now receive protection comparable to that of desktop users, and anti-phishing entities now better protect against some of the cloaking techniques I tested. I propose a number of additional improvements that could further strengthen the ecosystem, and I will freely release the PhishFarm framework to interested researchers and security organizations for continued testing of anti-phishing systems and potential adaptation for measuring variables beyond just cloaking. Thus, the contributions of this chapter are as follows:

- A controlled empirical study of the effects of server-side request filtering (*cloaking*) on blacklisting coverage and speed in modern desktop and mobile browsers.
- A reusable, automated, scalable, and extensible framework for carrying out my experiments.
- Identification of actionable real-world limitations in the current anti-phishing ecosystem.

- Disclosures to anti-phishing entities which led to enhancements to blacklisting infrastructure, including more reliable phishing protection in the mobile versions of Chrome, Safari, and Firefox.

## 4.2 Background

Phishing attacks are ever-evolving in response to ecosystem standards and may include innovative components that seek to circumvent existing mitigations. A current trend (mimicking the wider web) is the adoption of HTTPS by phishing websites, which helps them avoid negative security indicators in browsers and may give visitors a false sense of security [61, 137]. At the end of 2017, over 31% of all phishing websites reported to the APWG used HTTPS, up from less than 5% a year prior [5]. Another recent trend is the adoption of redirection links, which allows attackers to distribute a link that differs from the actual phishing landing page. Redirection chains commonly consist of multiple hops [121, 123], each potentially leveraging a different URL shortening service or open redirection vulnerability [25]. Notably, redirection allows the number of unique phishing links being distributed to grow well beyond the number of unique phishing websites, and such links might thus better slip past spam filters or volume-based phishing detection [136]. Furthermore, redirection through well-known URL shortening services such as *bit.ly* may better fool victims [19], though it also allows the intermediaries to enact mitigations.

Ultimately, phishers cleverly attempt to circumvent existing controls in an effort to maximize the effectiveness of their attacks. The anti-phishing community should seek to predict such actions and develop new defenses while ensuring that existing controls remain resilient.

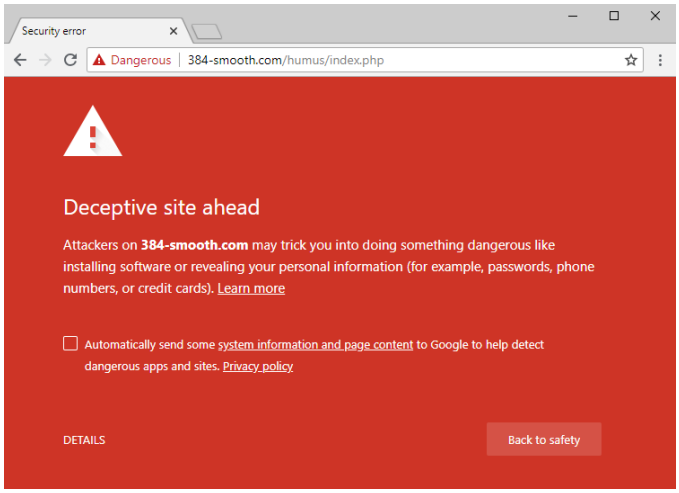



Figure 4.1: Phishing warning in Google Chrome 67.

### *4.2.1 Browser Blacklists*

Native browser blacklists are a key line of defense against phishing and malware websites, as they are enabled by default in modern desktop and mobile web browsers and thus automatically protect even unaware users against threats known to the blacklist provider. When a user tries to visit a phishing website that has been blacklisted, a prominent warning will be shown in place of the phishing content, thus protecting the user from the attack. Figure 4.1 depicts what the user might see when trying to visit a blacklisted website in Google Chrome (other browsers display similar warnings). In addition to the user-facing warnings found in browsers, blacklists have extensive backend infrastructure which detects and classifies phishing websites.

Today's major web browsers are protected by one of three different providers: Google Safe Browsing, Microsoft SmartScreen, or Opera. While Google Safe Browsing (GSB) and SmartScreen are well-documented standalone blacklists, Opera relies on third-party partners including PhishTank and Netcraft [111] to provide feeds of malicious websites.

There exist two other major blacklists with a lower global market share but which are prominent in specific countries [95]: *Tencent Security*, which protects the QQ browser (and Safari within China) [124], and *Yandex Safe Browsing*, which protects the Yandex browser [140]. In my experiments, I do not consider these blacklists, nor do I consider other anti-phishing systems which are not enabled by default in browsers, such as those only accessible via a browser plugin, third-party API, or antivirus software [101]. However, my methodology could be applied to evaluate these alternatives.

To the best of my knowledge, certain less prominent browsers, such as UC Browser, Samsung Internet, and Android Browser do not currently include native anti-phishing protection.

### *4.2.2 Browser Blacklist Operation*

In the absence of third-party security software, once a phishing message reaches and fools a potential victim, browser blacklists are the only technical control that stands between the victim and the display of the phishing content. Studies have shown that blacklists are highly effective at stopping a phishing attack whenever a warning is shown [114]. Such warnings are prominent but typically appear *after* the blacklist operator's backend web crawling infrastructure verifies the attack (some blacklists also leverage browser-based heuristics [114]). Thus, blacklists are inherently a *reactive* mitigation.

Blacklists face a difficult challenge: they must operate quickly and minimize false positives (to avoid disrupting legitimate websites) without allowing malicious content to go undetected. We know that blacklists are effective when they function as intended [99], but due to their reactive nature, they can be exploited through evasion techniques within phishing kits. If timely blacklisting does not occur, many users can be left vulnerable to attacks until (slower) secondary mitigations such as take-down occur [1, 89].

| *Browser Name* | *Phishing Blacklist Provider* | *Est. Market Share (Worldwide) [119, 120]* | |
|---|---|---|---|
| | | *7/2017* | *7/2018* |
| **Desktop Web Browsers** | | | |
| **Google Chrome*** | *Google* | 63.48% | 67.60% |
| **Mozilla Firefox*** | *Safe Browsing* | 13.82% | 11.23% |
| **Safari*** | *(GSB)* | 5.04% | 5.01% |
| **Internet Explorer (IE)*** | Microsoft | 9.03% | 6.97% |
| **Microsoft Edge*** | SmartScreen | 3.95% | 4.19% |
| **Opera*** | Opera | 2.25% | 2.48% |
| **Others** | Varies | 2.43% | 2.52% |
| **Mobile Web Browsers** | | | |
| **Google Chrome*** | GSB | 50.07% | 55.98% |
| **Safari*** | GSB | 17.19% | 17.70% |
| **UC Browser** | None | 15.55% | 12.49% |
| **Samsung Internet** | None | 6.54% | 5.12% |
| **Opera* (non-mini)** | Opera | 5.58% | 4.26% |
| **Android Browser** | None | 3.41% | 1.95% |
| **Mozilla Firefox*** | GSB | 0.06% | 0.31% |
| **Others** | Varies | 1.60% | 2.19% |

Table 4.1: Overview of the market share and blacklist providers of major web browsers (* denotes browsers tested in this chapter).

Just as users rely on these blacklists to stay safe, security organizations rely on them as an early part of their incident response process. If these blacklists fail to block malicious content in a timely and effective manner, organizations may remain defenseless in the short term as users fall victim to the attacks. On the other hand, when blacklisting succeeds, users remain protected while security organizations work to enact their more permanent anti-phishing mitigations.

### 4.2.3 *Detecting Phishing*

The distribution phase of phishing attacks is inevitably noisy, as links to each phishing website are generally sent to a large set of potential victims. Thus, the thousands of phishing attacks reported each month easily translate into messages with millions of recipients [78]. Detection can also occur during the preparation and exfiltration stages. As artifact trails propagate through the security community, they can be used to detect and respond to attacks. Detection methods include spam e-mail classification [32, 64], analyzing underground tools and drop zones [56], identifying malicious hostnames through passive DNS [11], URL and content classification [71, 136, 138, 142], malware scanning by web hosts [17], monitoring domain registrations [53] and certificate issuance [31], and receiving direct reports [1]. All of these potential detection methods can result in reports which may be forwarded to anti-phishing entities that power blacklists [114].

For users to be properly protected from phishing, malicious content must be detected in a timely manner so that it can be blacklisted or otherwise mitigated in the early stages of an attack. I elaborate on the importance of timely mitigation in Chapter 6.

### 4.2.4 *Cloaking*

In the previous chapter, I found that phishing kits commonly use server-side directives to *filter* (i.e., block or turn away) unwanted (i.e., non-victim) traffic, such as search engine bots, anti-phishing crawlers, security researchers, or users in locations that are incompatible with the phishing kit [102]. Attributes such as the visitor's IP address, hostname, user agent, or referring URL are leveraged to implement these filtering techniques.

Similar approaches, known as *cloaking*, have historically been used by malicious actors to influence search engine rankings by displaying different web content to bots than human visitors [58]. Users follow a misleading search result and are thus tricked into visit-

ing a website that could contain malware or adware. Relatively little research has focused on cloaking outside of search engines; my experiment in Section 4.3 is the first controlled study, to the best of my knowledge, that measures the impact of cloaking within phishing attacks. Because cloaking has the potential to evade (i.e., slow or prevent) the mitigation provided by blacklists, it warrants further scrutiny.

## 4.3 Experimental Methodology

My primary research goal in this chapter is to measure how cloaking affects the occurrence and speed of blacklisting of phishing websites within browsers. On a technical level, cloaking relies on filtering logic that restricts access to phishing content based on metadata from an HTTP request. As we saw in the previous chapter, filtering is widely used in phishing kits [102]; attackers aim to maximize their return on investment by only showing the phishing content to victims rather than anti-abuse infrastructure. If a phishing kit suspects that the visitor is not a potential victim, a 404 "not found", 403 "forbidden", or 30x redirection response code [40] may be returned by the server in lieu of the phishing page. Attackers could also choose to display benign content instead.

Prior studies of browser blacklists have involved observation or honeypotting of live phishing attacks [51, 99, 114], but these tests did not consider cloaking. It is also difficult to definitively identify cloaking techniques simply by observing live websites. My experimental design addresses this limitation.

### *4.3.1 Overview*

At a high level, I deploy my own (sterilized) phishing websites on a large scale, report their URLs to anti-phishing entities, and make direct observations of blacklisting times across major web browsers. I divide the phishing websites into multiple batches, each of which targets a different entity. I further sub-divide these batches into smaller groups

with different types of cloaking. Once the websites are live, I report their URLs to the anti-phishing entity being tested, such that the entity sees a sufficient sample of each cloaking technique. I then monitor the entity's response, with my primary metric being the time of blacklisting relative to the time I submitted each report. I collect secondary metrics from web traffic logs of each phishing website. My approach is thus empirical in nature, however it is distinct from prior work because I fully control the phishing websites in question, and, therefore, have ground truth on their deployment times and cloaking techniques.

All the phishing websites that I used for all of my experiments spoofed the *PayPal.com* login page (with permission from PayPal, Inc.) as it appeared in mid-2017 (I discuss the hosting approach in Section 4.4.3). My phishing websites each used new, unique, and previously-unseen *.com* domain names, and were hosted across a diverse set of IP addresses spanning three continents. As part of my effort to reliably measure the time between my report and browser blacklisting for each website, I never reported the same phishing website to more than one entity, nor did I re-use any domains. To summarize, each experiment proceeded as follows:

1. Selecting a specific anti-phishing entity to test.
2. Deploying a large set of new, previously-unseen PayPal phishing websites with desired cloaking techniques.
3. Reporting the websites to the entity.
4. Measuring if and when each website becomes blacklisted across major web browsers.

I split my experiments into two halves: preliminary testing of ten anti-phishing entities (mid-2017) and full follow-up testing of five of the best-performing entities (mid-2018).

The latter test involved a large sample size designed to support statistically significant inferences. In between the two tests, I disclosed my preliminary findings to key entities to allow them to evaluate the shortcomings that I identified. I discuss my approach in more detail in the following sub-sections and present the results in Section 4.5.

### Choosing Web Browsers to Evaluate

I strove to maximize total market share while selecting the browsers to be tested. In addition to traditional desktop platforms, I wanted to measure the extent of blacklisting on mobile devices—a market that has grown and evolved tremendously in recent years [131]. I thus considered all the major desktop and mobile web browsers with native anti-phishing blacklists, as listed in Table 4.1.

### Filter Types

I chose a set of request filtering techniques based on the high-level cloaking strategies from my study of *.htaccess* files in phishing kits that I presented in Chapter 3 [102]. Exhaustively measuring every possible combination of request filters was not feasible while maintaining a large sample size. Therefore, I chose a manageable set of filtering strategies that I felt would be effective in limiting traffic to broad yet representative groups of potential victims while remaining simple (for criminals) to implement and drawing inspiration from the techniques found in the wild [58]. Table 4.3 summarizes my filter selections.

It would not be responsible for me to disclose the exact conditions required of each filter, but I can discuss them at a high level. *Filter A* served as my control group; my expectation was for every website in this group to be blacklisted at least as quickly as other websites. *Filter B* sought to study how well mobile-only phishing websites are blacklisted, coinciding with the recent uptick in mobile users and phishing victims [5, 119]. *Filters C* and *D* focus specifically on desktop browsers protected by GSB, which span the majority

of desktop users today. I also included geolocation, which while not as frequent in the wild as other cloaking types [102], is highly effective due to low detectability and characteristics of *spearphishing*. A secondary motivation was to see how well entities other than GSB protect this group. *Filter E* is the most elaborate, but it also directly emulates typical real-world phishing kits. It is based on top *.htaccess* filters from a large dataset of recent phishing kits [102]. This filter seeks to block anti-phishing entities by hundreds of IP addresses, hostnames, referrers, and user agents. Finally, *Filter F* only displays phishing content if the client browser can execute JavaScript, which may defeat simple script-based web crawlers. This filter is common in modern search engine cloaking [58]. Although today's phishing kits tend to favor straightforward filtering techniques (namely approaches such as *Filter E* or geolocation), growing sophistication and adoption of cloaking (and other types of evasion) is technically feasible and to be expected as a risk to the anti-phishing ecosystem.

### Tested Entities

In addition to the blacklist operators themselves, major clearinghouses, and PayPal's internal anti-phishing system, I wanted to test as many of the other types of anti-phishing entities discussed in Section 2.4 as possible. I started with a recently-published list of entities commonly targeted for evasion by phishers [102]. I then made selections from this list, giving priority to the more common (and thus potentially more impactful) entities in today's ecosystem. I had to exclude some entities of interest, as not all accept direct external phishing reports and thus do not fit the experimental design. Also, I could not be exhaustive as I had to consider the domain registration costs associated with conducting each experiment. Table 4.2 summarizes my entity selections.

| *Entity (Report Location)* | *Report Type* | *URLs* |
|---|---|---|
| **Full + Preliminary Tests** | | |
| APWG (reportphishing@apwg.org) | E-mail | 40 (prelim.)<br>396 (full)<br>per entity |
| Google Safe Browsing ([44]) | Web | |
| Microsoft SmartScreen ([84]) | Web | |
| PayPal (spoof@paypal.com) | E-mail | |
| PhishTank (phish-{username}@phishtank.com) | E-mail | |
| **Preliminary Tests Only** | | |
| ESET ([34]) | Web | 40<br>per entity |
| Netcraft ([97]) | Web | |
| McAfee ([80]) | Web | |
| US CERT (phishing-report@us-cert.gov) | E-mail | |
| WebSense (asa@websense.com) | E-mail | |
| **10 Entities Total** | | **2,380 URLs** |

Table 4.2: Entities targeted by my experiments.

### Reporting

When the time came for us to report my phishing websites to each entity being tested, I would submit the website URLs either via e-mail or the entity's web interface (if available and preferred by the entity). In all cases, I used publicly-available submission channels; I had no special agreements with the entities, nor did they know they were being tested. In the case of the web submission channel, I simply used a browser to submit each phishing website's exact URL to the entity. E-mail reports were slightly more involved, as the industry prefers to receive attachments with entire phishing e-mails in lieu of a bare URL. I thus used PayPal-branded HTML e-mail templates to create lookalike phishing messages. Each message contained a random customer name and e-mail address (i.e., of a hypotheti-

| *Cloaking Filter Name* | | *HTTP Request Criteria* |
|---|---|---|
| **A** | Control | Allow all |
| **B** | Mobile Devices | Allow if user agent indicates:<br>Android or iOS |
| **C** | US Desktop<br>GSB Browsers | Allow if IP country (is/is not) US and user agent indicates:<br>Chrome, Firefox, or Safari; and<br>Windows, Macintosh, or Linux; and<br>not Opera, IE, or Edge; and<br>not iOS or Android |
| **D** | Non-US Desktop<br>GSB Browsers | |
| **E** | Block Known<br>Security Entities | Allow if user agent, referrer, hostname, and IP:<br>not known to belong to a security entity or bot |
| **F** | "Real" Web<br>Browsers | Allow all; content retrieved asynchronously<br>during JavaScript onload event |

Table 4.3: Cloaking techniques used by my phishing websites.

cal victim) within one of many body templates. I sent e-mails from unique accounts across five security-branded domains under my control.

Reports of real-world phishing websites might be submitted in an identical manner by agents on behalf of victim brands. My reporting approach is thus realistic, though many larger victim organizations contract the services of enterprise security firms, which in turn have private communication channels to streamline the reporting process.

### *4.3.2 Preliminary Tests*

My preliminary testing took place in mid-2017 and carried out the full experimental approach from Section 4.3.1 on a small scale. I will now detail the execution of each test.

One of my key goals was to minimize confounding effects on the experimental results. In other words, I did not want any factors other than the report submission and cloaking

strategy to influence the blacklisting of my phishing websites. As part of this, I wanted to secure a large set of IP addresses, such that no anti-phishing entity would see two of my websites hosted on the same IP address. With the resources available for this research, I was able to provision 40 web servers powered by Digital Ocean, a large cloud hosting provider [28] (I informed Digital Ocean about this research to ensure the servers would not be taken down for abuse [89]). Each server had a unique IP address hosted in one of the host's data centers in Los Angeles, New York, Toronto, Frankfurt, Amsterdam, or Singapore. The batch size was thus 40 phishing websites per preliminary test, for a total of 400 websites across the ten entities being tested. Within each batch, six websites used filter types *A* and *F*, while seven used filter types *B* through *E*.

I also wanted to mimic actual phishing attacks as closely as possible. I studied the classification of phishing URL types submitted to the Anti-phishing Working Group's eCrime Exchange in early 2017 and crafted the URLs for each of the 40 phishing websites while following this distribution as closely as possible [41, 102]. I registered only *.com* domains for the URLs, as the *.com* TLD accounts for the majority of real-world phishing attacks. In addition, I chose GoDaddy as the registrar, which is among the most-abused registrars by real phishers [4]. Furthermore, to prevent crawlers from landing on my phishing websites by chance (i.e., by requesting the bare hostname), paths were non-empty across all of the URLs.

Using URLs that appear deceptive is a double-edged sword: while it allows us to gain insight into how various URL types are treated by entities, it is also a factor that may skew blacklisting speed. However, I decided to proceed with this in mind as the purpose of the preliminary tests was to *observe* more than *measure*, given the use of a relatively small sample size per batch. Table 4.4 shows the distribution of URL types and filters per batch of websites.

I registered the required domain names and finalized the configuration of my infrastructure in May 2017. In July, I started the experiments by systematically deploying and reporting my phishing websites. For each day over the course of a ten-day period, I picked one untested entity at random (from those in Table 4.2), fully reported a single batch of URLs (over several minutes), and started monitoring blacklist status across each targeted web browser. The monitoring of each URL continued every ten minutes for a total of 72 hours after deployment. Over this period and for several days afterward, I also logged web traffic information to each of my phishing websites in an effort to study crawler activity related to each entity. Prior empirical tests found blacklists to show their weakness in the early hours of a phishing attack [114]. My 72-hour observation window allows us to study blacklist effectiveness during this critical period while also observing slower entities and potentially uncovering new trends.

### *4.3.3 Responsible Disclosure*

My analysis of the preliminary tests yielded several security recommendations (discussed in Section 4.6). I immediately proceeded to disclose my findings to the directly impacted entities (i.e., browser vendors, the brand itself, and major blacklist operators) which I also intended to re-test. I held my first disclosure meeting with PayPal in August 2017. Following PayPal's legal approval, I also disclosed to Google, Microsoft, Apple, Mozilla, and the APWG in February 2018. Each meeting consisted of a detailed review of the entity's performance in my study, positive findings, specific actionable findings, and high-level comparisons to other entities. The disclosures were generally positively received and resulted in close follow-up collaboration with Google, Mozilla, and the APWG; this ultimately resulted in the implementation of effective blacklisting within mobile GSB browsers and general mitigations against certain types of cloaking. I clearly stated that I would repeat the experiments in 4-6 months and thereafter publish the findings.

| *Phishing URL Content* / *Sample URL* (*Type* [41, 102]) | *Qty.* | *Filters Used* |
|---|---|---|
| **Full Tests** | | |
| Non-deceptive (random) (V)<br>*http://www.florence-central.com/logician/retch/* | 396 | A (66), B (66), C (66), D (66), E (66), F (66) |
| **Preliminary Tests** | | |
| Non-deceptive (random) (V)<br>*http://receptorpeachtreesharp.com/cultivable/* | 10 | A (1), B (2), C (2), D (2), E (2), F (1) |
| Brand in Domain (IVa)<br>*http://www.https-official-verifpaypal.com/signin* | 6 | A (1), B (1), C (1), D (1), E (1), F (1) |
| Deceptive Domain (IVb)<br>*http://services-signin.com/login/services/account/* | 6 | |
| Brand in Subdomain (IIIa)<br>*http://paypal1.com.835anastasiatriable.com/signin* | 6 | |
| Deceptive Subdomain (IIIb)<br>*http://services.account.secure.lopezben.com/signin* | 6 | |
| Deceptive Path (II)<br>*http://simpsonclassman.com/paypa1.com/signin* | 6 | |

Table 4.4: URL and filter distribution for each experiment.

I did not immediately disclose to the blacklists powering Opera as I had not originally expected to have the resources to re-test a fifth entity. Additionally, given lesser short-term urgency with respect to the remaining entities (and in the absence of close relationships with them), at the time I felt it would be more impactful to disclose to them the preliminary findings *alongside* the full test findings. This approach ultimately allowed us to better guide my recommendations by sharing deeper insight into the vulnerabilities within the broader ecosystem. After the completion of the full tests, I reached out the all remaining entities via e-mail; all but PhishTank and Opera responded and acknowl-

edged receipt of a textual report containing my findings. US CERT and ESET additionally followed up for clarifications once thereafter.

### *4.3.4 Full-scale Tests*

I believed that key ecosystem changes resulting from my disclosures would still be captured by the original experimental design. Thus, I did not alter the retesting approach beyond increasing the scale to enable a more representative sample size. In addition, rather than using the URL distribution from the preliminary experiments, I solely used non-deceptive paths and hostnames (i.e., with randomly-chosen English words) in order to remove URL classification as a possible confounding factor in my evaluation of cloaking.

I registered the required domains in May 2018 and initiated sequential deployment of the full-scale tests in early July. I re-tested all entities to which I disclosed. As my budget ultimately allowed for an additional entity, I decided to also include PhishTank for its promising performance in the preliminary test. Each of the resulting five experiment batches consisted of 396 phishing websites, evenly split into six groups for each cloaking technique. My reporting method did not change, though I throttled e-mailing such that the reports were spread over a one-hour period. Reporting through Google and Microsoft's web interfaces spanned a slightly longer period of up to two hours due to the unavoidable manual work involved in solving required CAPTCHA challenges.

### *4.3.5 Sample Size Selection*

For the full tests, I chose a sample size of 384 phishing websites for each entity. My goal was to obtain a power of 0.95 at the significance level of 0.05 in a one-way independent ANOVA test, which, for each entity, could identify the presence of a statistically significant difference in mean blacklisting speed between the six cloaking filters. Based on Cohen's recommendation [20] and my preliminary test results, I assumed a *medium* effect

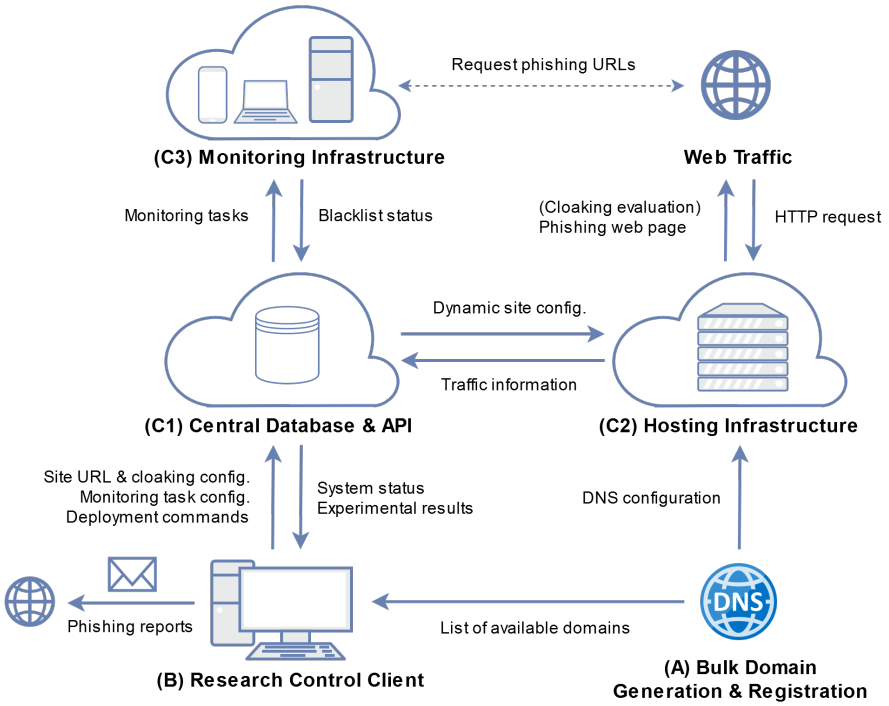



Figure 4.2: PhishFarm framework architecture.

size ($f$) of 0.25. I added 12 websites (2 per filter) to each experiment to serve as backups in case of unforeseen technical difficulties. All websites ultimately delivered 100% uptime during deployment, thus I ended up with an effective sample size of 396 per experiment. While the assumption of *f=0.25* introduced some risk into the experimental design, I accepted it given the high cost of registering a new *.com* domain for each website.

## 4.4 PhishFarm Testbed Framework

To be able to execute my experimental design at scale, I designed *PhishFarm*: a comprehensive framework for deploying phishing websites, reporting them to anti-abuse entities, and measuring blacklist response. The framework operates as a web service and satisfies three important requirements: automation, scalability, and reliability. I eliminated as many manual actions as technically feasible to ensure that the difference between launching hundreds and thousands of websites was only a matter of minutes. Actual website

deployment can thus happen instantly and on-demand. Apart from up-front bulk domain registration (4.4.1) and reporting phishing through a web form, all framework components feature end-to-end automation.

The framework consists of five interconnected components as shown in Figure 4.2: bulk domain registration and DNS setup, stateless cloud web servers to display the phishing websites, an API and database that manages configuration, a client that issues commands through the API, and a monitoring system that regularly checks in which browsers each phishing is blacklisted. In total, I wrote over 11,000 lines of PHP, Java, and Python code for these backend components. I extensively tested each component—in particular, the hosting and monitoring infrastructure—to verify its correct operation.

### *4.4.1 Domain and DNS Configuration*

A core component of any phishing website is its URL, which consists of a hostname and path [41]. The framework includes a script to automatically generate hostnames and paths per the experimental design (Ⓐ). I then manually register the required domain names in bulk and point them to my cloud hosting provider's nameservers. Finally, I automatically create DNS records such that the domains for each experiment are evenly spread across the IP addresses under my control. I set up wildcard CNAME records [70] such that I could programmatically configure the subdomain of each phishing website on the server-side. In total, registration of the 400 preliminary domains took about 15 minutes, while registration of the 1,980 domains for the full tests took 30 minutes; apart from the domain registration itself, no manual intervention is needed.

### *4.4.2 Research Client*

I implemented a cross-platform client application (Ⓑ) to control the server-side components of the framework and execute my research. This client enables bulk configu-

ration of phishing websites and cloaking techniques, bulk deployment of experiments, automated e-mail reporting, semi-automated web reporting, monitoring of status, and data analysis.

### *4.4.3 Server-side Components*

The server-side components in my framework display the actual phishing websites based on dynamic configuration. They are also responsible for logging request data for later analysis and monitoring blacklist status.

#### Central Database and API

At the heart of the framework is a central API (C1) that serves as an interface to a database with the phishing website configuration and system state. For each website, the database maintains attributes such as date activated, reported, and disabled; blacklisting status; website and e-mail HTML templates; server-side and JavaScript request filtering code; and access logs. I interact with this API via the client whenever I need to define new websites or deploy websites as part of an experiment. All traffic to and from the API is encrypted.

#### Hosting Infrastructure

My hosting infrastructure (C2) consists of Ubuntu cloud servers which run custom intermediate software on top of Apache. The intermediate software captures all requests and enables dynamic cloaking configuration and enhanced traffic logging. Each server obtains all of its configuration from the API, which means that the number of servers can flexibly be chosen based on testing requirements (40 in this case). When an HTTP request is received by any server, the server checks the request URL against the list of live phishing websites from the API, processes any cloaking rules, responds with the intended

content, and logs the request. To quickly handle incoming requests, the system state is cached locally on each server to minimize API queries.

I served all my phishing websites over HTTP rather than HTTPS, as this was typical among real-world phishing websites at the time I designed the preliminary tests [4]. Although the PhishFarm framework supports HTTPS, I did not wish to alter the experimental design between tests. Both approaches generate potential artifacts that might benefit the anti-phishing ecosystem: unencrypted websites allow network-level packet sniffing, while encrypted websites leave an evidence trail at the time a certificate is issued [31]. I mitigated the former risk to a large extent through a stealthy design of the monitoring system, as discussed in more detail in the following section.

My framework supports arbitrary, brand-agnostic web page content to be displayed for each phishing website. For my experiments, I hosted all resources (e.g., images and CSS) locally on each server to avoid confounding detection through external web requests to these files. In the wild, I have observed that sophisticated phishing kits follow a similar strategy to reduce detectability, though today's typical kits merely embed resources from the legitimate website.

### Monitoring Infrastructure

The purpose of the monitoring system (C3) is to identify, at a fine granularity, how much time elapses before a reported phishing website is blacklisted (if at all). To obtain a clear picture of the early hours of blacklists' response, I configured the monitoring system to access each live phishing URL at least once every ten minutes in *each* target desktop browser (the shortest feasible interval with the available hardware). I checked the blacklist status of each URL by analyzing a screenshot of the respective browser window; this is similar to the approach taken by Sheng et al [114]. I chose this approach for its universal applicability and lack of dependencies on browser automation libraries, which commonly

force-disable phishing protection [2]. For my purposes, it sufficed to check if the dominant color in each image [112] was close to red (red is used in the phishing warning messages of all the browsers I considered). Although this image-based approach proved reliable and can be fully automated, it does not scale well because each browser requires exclusive use of a single system's screen.

To satisfy the scalability requirement, the monitoring system follows a distributed architecture with multiple autonomous nodes that communicate with the API; each node is a virtual machine (VM) that runs on a smaller set of host systems. Software on each VM points a browser to the desired URL via command line and sends virtual keystrokes, as needed, to close stale tabs and allow for quick page loads. During the experiments, I used three types of nodes: Mac OS X 10.12 VMs with Chrome, Safari, Opera, and Firefox; Windows 10 VMs with Edge and Internet Explorer (IE) 11; and Windows 8.1 VMs with IE 11. In total, the full experiments required 31 desktop nodes across four host machines (collectively with 18 Intel Core i7 CPU cores and 96 Gb of RAM) to deliver the required level of monitoring performance and redundancy. Each node informs the API of the browsers it supports; it then awaits a command consisting of a set of URLs for a target browser, and in real-time, reports the monitoring results back to the API. I freshly installed the latest stable version of each browser at the time of each test and kept default security settings (or, in the case of IE and Edge, recommended settings when prompted).

I was unable to obtain such a large number of mobile devices, and official emulators at the time would force-disable blacklist warnings. Thus, I only tested mobile browsers hourly and relied on my observation that their behavior was tied to the behavior of their desktop counterparts. I used a physical Samsung Galaxy S8 and Google Pixel phone to test mobile Chrome, Firefox, and Opera; and an iPhone 7 (preliminary) or 8 (full) to test mobile Safari. A future version of the framework could be improved to leverage Android

VMs, in lieu of physical or emulated devices, to perform monitoring similar to that of desktop browsers.

Lastly, I wanted to ensure that the large number of requests from the monitoring system did not affect the blacklist status of my phishing websites (i.e., by triggering heuristics or network-level analysis [114]). Therefore, rather than displaying the phishing content in the monitoring system's browsers, I simply displayed a blank page with an *HTTP 200* (OK) status code [40]. Although I had the option of querying the GSB API [48] directly as a supplement to empirically monitoring the browsers it protects, I chose not to do so for the same reason. Finally, the monitors were connected to the Internet using an anonymous VPN service in an effort to bypass any potential institution- and ISP-level sniffing.

### *4.4.4 Ethical and Security Concerns*

Throughout my experiments, I was careful in ensuring that no actual users would visit (let alone submit credentials to) my phishing websites: I only ever shared the website URLs directly with anti-phishing entities, and I sterilized the login forms such that passwords would not be transmitted in the event of form submission. In the hands of malicious actors with access to the necessary hosting and message distribution infrastructure, and with malicious modifications, my framework could potentially be used to carry out real and evasive phishing attacks on a large scale. I will thus not release the framework as open-source software; however, it will be made available to vetted security researchers.

Another potential concern is that testing such as that done by PhishFarm could degrade the response time of live anti-phishing systems or negatively impact the data samples used by their classifiers. Given the volume of live phishing attacks today [5], I do not believe that my experiments carried any adverse side effects, and the entities to which I disclosed did not raise any concerns regarding this. With respect to classifiers based on machine learning, an effective methodology has already been proposed to ensure that

high-frequency phishing websites do not skew training samples [136]. Nevertheless, it would be prudent for any party engaged in long-term or high-volume testing of this nature to first consult with any targeted entities.

## 4.5 Experimental Results

PhishFarm proved to deliver reliable performance over the course of my tests and executed the experimental methodology as designed. Following the completion of all tests, I had collected timestamps of when blacklisting occurred, relative to the time I reported the website, for each of my phishing websites in each of the desktop and mobile browsers tested. This totaled over 20,000 data points across the preliminary and full tests, and had added dimensionality due to the various entities and cloaking techniques tested. Table 4.5 shows the breakdown of crawler and blacklisting activity on my phishing websites. Overall, I found that websites with cloaking were both slower and less likely to be blacklisted than websites without cloaking: in the full tests, just 23.0% of the websites with cloaking (which were crawled) ended up being blacklisted in at least one browser—far fewer than the 49.4% websites without cloaking which were blacklisted. Cloaking also slowed the average time to blacklisting from 126 minutes (for websites without cloaking) to 238 minutes.

However, closer scrutiny is required to make insightful conclusions about the conditions under which cloaking is effective, as I found that each anti-phishing entity exhibited distinctive blacklisting of different cloaking techniques alongside varying overall speed. For example, the mobile-only *Filter B* showed 100% effectiveness against blacklisting across all entities. On the other hand, the JavaScript-based *Filter F* was 100% effective for some entities, delayed blacklisting for others, but was, in fact, more likely to be blacklisted than *Filter A* by others still. In many cases, there was also a lack of blacklisting of *Filter A* websites (i.e., those *without* cloaking). Entities—in particular, the clearinghouses—

| | Phishing Websites Deployed | | Crawler(s) Attempted Retrieval | | Successful Crawler Retrievals | | Crawled Websites Blacklisted | | Mean Time Before 1st Blacklisting | | Mean Page Loads per Website | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *w/ Cloaking* | *w/o* | *w/* | *w/o* | *w/* | *w/o* | *w/* | *w/o* | *w/* | *w/o* | *w/* | *w/o* |
| **Prelim.** | 340 | 60 | 294 (86.5%) | 51 (85.0%) | 156 (53.1%) | 49 (96.1%) | 124 (42.2%) | 42 (82.4%) | 173 min. | 79 min | 80 | 464 |
| **Full** | 1650 | 330 | 1333 (80.8%) | 271 (82.7%) | 818 (61.4%) | 264 (97.4%) | 306 (23.0%) | 134 (49.4%) | 238 min. | 126 min. | 162 | 334 |

Table 4.5: Overview of crawler and blacklisting activity across all experiments.

at times failed to ultimately blacklist a reported website despite the extensive crawling activity. Although it is possible that the direct reporting methodology led to some websites being ignored altogether, the exceptional performance of GSB with respect to *Filter A* shows that a very high standard is realistic. I detail each entity's behavior in Section 4.5.6.

To provide meaningful measurements of entity performance with respect to different browsers and cloaking filters, I propose a scoring system in Section 4.5.2 which seeks to capture both the response time and the number of websites blacklisted by each anti-phishing entity for each browser and/or filter. In addition, I summarize my findings visually in Figures 4.3 and 4.4 by plotting the cumulative percentage of websites blacklisted over time, segmented by each browser or by each cloaking technique, respectively. Lastly, I analyze web traffic logs to make observations about the distinctive behavior of each anti-phishing entity. Although much of my analysis focuses on the results of the large-scale full tests, I also make comparisons to the preliminary test data when appropriate.

### *4.5.1 Crawler Behavior*

Of all websites that I launched during the preliminary and full tests, most (81.9%) saw requests from a web crawler, and a majority of websites therein (66.0%) were successfully retrieved at least once (i.e., bypassing cloaking if in use). In a handful of cases, my reports were ignored by the entities and thus resulted in no crawling activity, possibly due to

volume or similarity to previous reports; I mitigated this risk through the large sample size and discuss it in more detail in Section 4.7. The distribution of crawler hits was skewed left, characterized by a small number of early requests from the entity to which I reported, followed by a large stream of traffic from it as well as other entities. Importantly, different cloaking techniques showed no significant effect on the time of the first crawling *attempt*; the median response time ranged from 50 to 53 minutes, from the time of reporting, for all filter types.

During the full experiments, *only* websites that were crawled were ultimately blacklisted. Generally, the crawling also had to result in successful retrieval of the phishing content for blacklisting to occur, though in 10 cases (all in the GSB experiment with *Filter D*), a website would be blacklisted despite a failed retrieval attempt; possible factors for this positive phishing classification are described in prior work [136].

### *4.5.2 Entity Scoring*

For each entity tested, let $U$ be the set of phishing URLs reported to the entity, let $B$ be the set of browsers monitored, let $T$ be the set of observed blacklisting times, let $F$ be the set of cloaking filters used, and let $MS$ denote the worldwide browser market share. Additionally, I define the value of the function *accessible(b, f)* to be true if and only if a phishing website with filter $f$ is designed to be accessible in browser *b*.

For each URL-browser combination in $B \times U$, per Formula 4.1, I define a normalized performance score $S_{URL_b}$ on the range $[0, 1]$, where 0 represents no blacklisting and 1 represents immediate blacklisting relative to the time reported. The score decays linearly over the 72-hour observation window (e.g., a website blacklisted after 36 hours would have $S_{URL_b} = 0.5$). I take the average of all these URL-browser scores for each browser-filter combination, as $S_{bf}$, per Formula 4.2.

$$\forall\ b,\ URL \in B \bigtimes U,\ S_{URL_b} = \tag{4.1}$$

$$\begin{cases} 1 - \frac{T_{URL\ blacklisted_b} - T_{URL\ reported}}{T_{window}} & blacklisted\ in\ b \\ 0 & otherwise \end{cases}$$

$$\forall\ b,\ f \in B \bigtimes F,\ S_{bf} = \sum_{URL,\ F_{URL}=f} \frac{S_{URL_b}}{n} \tag{4.2}$$

$$\forall\ b \in B,\ S_b = \sum_{f,\ accessible(b,\ f)} \frac{S_{bf}}{n} \tag{4.3}$$

$$S = \sum_b \frac{MS_b}{MS_B} \cdot S_b \tag{4.4}$$

Table 4.6: Formulas for aggregate entity scores (per test).

I further aggregate the $S_{bf}$ scores to meaningfully summarize the protection of each browser by each entity: the browser score $S_b$, as per Formula 4.3, is the average of all $S_{bf}$ for a given browser *b*, but limited to filters $f$ *accessible* in *b*. To gauge the blacklisting of each cloaking filter, I report $PB_f$ as the raw proportion of websites blacklisted in at least one browser for each respective filter. Note that $PB_f$ will always be greater than or equal to any $S_{bf}$ because the former is not reduced by slow speed; I thus additionally report $TB_f$, the average time to blacklisting for each filter $f$ (in minutes).

To capture the overall real-world performance of each entity, I average all $S_b$, weighted by the market share of each browser, to produce $S$, as per Formula 4.4. The scores $S$ allow us to directly and efficiently compare the performance between entities, and would be useful in modeling long-term trends in future deployments of *PhishFarm*. I also calculate the proportion of all websites crawled, $C$, to illustrate the entity's response effort.

I present the aforementioned aggregate scores in Table 4.7 for all entities in the full tests. Indeed, the abundance of *0* (and *near-0*) scores was disappointing and representative of multiple ecosystem weaknesses which I discuss in the following sections. Scores for the preliminary tests are found in Table A.1 in Section A. Note that because Chrome, Firefox,

and Safari showed nearly identical scores across all experiments, I simplify the table to report the highest respective score under the *GSB* heading. I make a similar simplification for IE 11 on Windows 10 and 8.1.

I do not separately report the performance of mobile browsers because I observed the behavior of mobile browsers to be directly related to their desktop counterparts. During the preliminary tests, mobile Firefox and Opera mirrored the blacklisting—and thus also the scores—of their desktop versions. Mobile Chrome and Mobile Safari showed no blacklist warnings *whatsoever* for any of my phishing websites and thus receive $S_b$ scores of *0*. During the full tests, the behavior of mobile Chrome, Safari, and Opera remained unchanged. Firefox stopped showing blacklist warnings, and its scores thus dropped to *0* (the *0* scores of mobile browsers represented a serious issue; this was corrected after I contacted Google and Mozilla after the full tests, as discussed in Section 4.6.1). I did not test mobile versions of Microsoft browsers because mobile IE is no longer supported; Edge for Android and iOS was released after I began testing.

Table 4.7: Aggregate entity blacklisting performance scores in the full tests (colors denote subjective assessment: green— good, yellow— lacking, red— negligible blacklisting).

| **GSB** | | *Filter A* | *Filter B* | *Filter C* | *Filter D* | *Filter E* | *Filter F* | $S_b$ | |
|---|---|---|---|---|---|---|---|---|---|
| $S_{bf}$ | *GSB* | 0.942 | 0 | 0.030 | 0.899 | 0.104 | 0.692 | 0.533 | |
| | *IE* | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| | *Edge* | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| | *Opera* | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| $PB_f$ | | 0.970 | 0 | 0.031 | 0.953 | 0.106 | 0.712 | **0.457** | $S$ |
| $TB_f$ *(min.)* | | 112 | N/A | 50 | 100 | 81 | 107 | 0.947 | $C$ |

| **SmartScreen** | | *Filter A* | *Filter B* | *Filter C* | *Filter D* | *Filter E* | *Filter F* | $S_b$ | |
|---|---|---|---|---|---|---|---|---|---|
| $S_{bf}$ | *GSB* | 0.005 | 0 | 0.005 | 0 | 0 | 0.009 | 0.004 | |
| | *IE* | 0.176 | 0 | 0.003 | 0 | 0.301 | 0.411 | 0.296 | |
| | *Edge* | 0.183 | 0 | 0.003 | 0 | 0.329 | 0.421 | 0.311 | |
| | *Opera* | 0 | 0 | 0.005 | 0 | 0 | 0 | 0 | |
| $PB_f$ | | 0.212 | 0 | 0.016 | 0 | 0.364 | 0.455 | **0.038** | $S$ |
| $TB_f$ | | 548 | N/A | 2889 | N/A | 391 | 298 | 0.649 | $C$ |

| **APWG** | | *Filter A* | *Filter B* | *Filter C* | *Filter D* | *Filter E* | *Filter F* | $S_b$ | |
|---|---|---|---|---|---|---|---|---|---|
| $S_{bf}$ | *GSB* | 0.563 | 0 | 0.356 | 0 | 0.777 | 0 | 0.339 | |
| | *IE* | 0.113 | 0 | 0 | 0 | 0.626 | 0 | 0.246 | |
| | *Edge* | 0.129 | 0 | 0 | 0 | 0.761 | 0 | 0.297 | |
| | *Opera* | 0.242 | 0 | 0.185 | 0 | 0.545 | 0 | 0.262 | |
| $PB_f$ | | 0.576 | 0 | 0.344 | 0 | 0.803 | 0 | **0.328** | $S$ |
| $TB_f$ | | 194 | N/A | 243 | N/A | 125 | N/A | 1 | $C$ |

| **PhishTank** | | *Filter A* | *Filter B* | *Filter C* | *Filter D* | *Filter E* | *Filter F* | $S_b$ | |
|---|---|---|---|---|---|---|---|---|---|
| $S_{bf}$ | *GSB* | 0.077 | 0 | 0 | 0 | 0 | 0.026 | 0.024 | |
| | *IE* | 0.096 | 0 | 0 | 0 | 0 | 0 | 0.026 | |
| | *Edge* | 0.085 | 0 | 0 | 0 | 0 | 0 | 0.028 | |
| | *Opera* | 0.074 | 0 | 0 | 0 | 0 | 0.024 | 0.032 | |
| $PB_f$ | | 0.106 | 0 | 0 | 0 | 0 | 0.136 | **0.025** | $S$ |
| $TB_f$ | | 386 | N/A | N/A | N/A | N/A | 2827 | 0.467 | $C$ |

| **PayPal** | | *Filter A* | *Filter B* | *Filter C* | *Filter D* | *Filter E* | *Filter F* | $S_b$ | |
|---|---|---|---|---|---|---|---|---|---|
| $S_{bf}$ | *GSB* | 0.133 | 0 | 0.149 | 0.052 | 0.198 | 0.167 | 0.104 | |
| | *IE* | 0.102 | 0 | 0.040 | 0 | 0.074 | 0.137 | 0.140 | |
| | *Edge* | 0.123 | 0 | 0.056 | 0.046 | 0.193 | 0.163 | 0.160 | |
| | *Opera* | 0.119 | 0 | 0.029 | 0 | 0.191 | 0.120 | 0.143 | |
| $PB_f$ | | 0.167 | 0 | 0.172 | 0.078 | 0.288 | 0.182 | **0.138** | $S$ |
| $TB_f$ | | 675 | N/A | 440 | 1331 | 1077 | 338 | 0.995 | $C$ |

### 4.5.3 *Preliminary vs. Full Tests*

I observed many core similarities when comparing the performance of the same entity between the preliminary and full tests. I also saw improvements related to the recommendations I shared during the disclosure meetings, in particular concerning the APWG's treatment of *Filters C* and *E*. Notably, during the full tests, crawler traffic to websites *with* cloaking increased by 44.7% relative to websites *without* cloaking, while the overall traffic volume also increased by 89.7%. I discuss all the entity-specific improvements in Section 4.5.6.

The comparison also revealed some surprises, however. The main experimental difference between the two sets of tests, apart from the sample size, was my exclusive use of random URLs in the full tests. On the other hand, the preliminary tests included a sampling of deceptive URLs. In the preliminary tests, I observed that Edge and IE were quick to display blacklist warnings for websites with certain deceptive URLs. In fact, many Type IV URLs (with domain names containing either the PayPal brand or deceptive keywords) saw proactive zero-hour warnings in these browsers without any prior crawler activity. Figure A.1b in Section A shows the effect of URL type on blacklisting in the preliminary tests. During the full tests, no phishing website was blacklisted unless it had previously been visited by a crawler. In the absence of deceptive URLs, I thus observed all the blacklists to be purely reactive; this, in turn, led to lower $S_b$ scores of Edge, IE, and websites with *Filter B*, and a lower overall score $S$ for SmartScreen.

Raw data from the preliminary tests can be found in Appendix A.

### 4.5.4 *Browser Performance*

Figure 4.3 shows the percentage of my phishing websites blacklisted over the course of all the full experiments, grouped by browser, but limited to websites intended to be

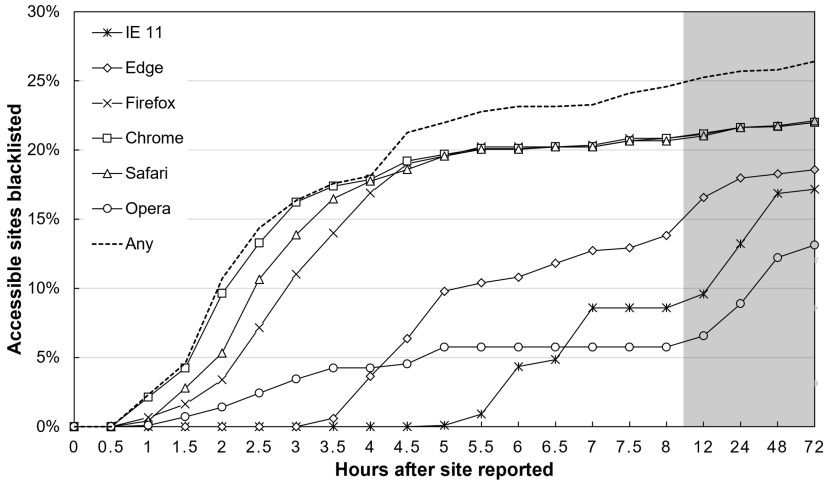



Figure 4.3: Blacklisting over time in each browser (full tests).

*accessible* in each respective desktop browser (i.e., *Filter B* was excluded for all and *Filters C* and *D* were excluded for IE, Edge, and Opera).

Chrome, Safari, and Firefox consistently exhibited the highest overall blacklisting speed and coverage, but were still far from blacklisting all cloaked websites. Opera generally outperformed the Microsoft browsers during the first four hours but was later overtaken for the remainder of the experimental period. In the absence of deceptive phishing URLs, Edge and IE simply lost their edge in the full tests; Figure A.1c in Section A shows their superior performance in the preliminary tests.

### Consistency Between Browsers

Chrome, Firefox, and Safari are all protected by the GSB blacklist; my tests confirmed these browsers do consistently display blacklist warnings for the same set of websites. However, during the full tests, Chrome displayed warnings up to 30 minutes earlier than Safari, and up to one hour earlier than Firefox; the warnings became consistent after the five-hour mark. Upon further investigation, I believe that this disparity was caused by different caching implementations of the *GSB Update API (v4)* in each browser (this API ensures

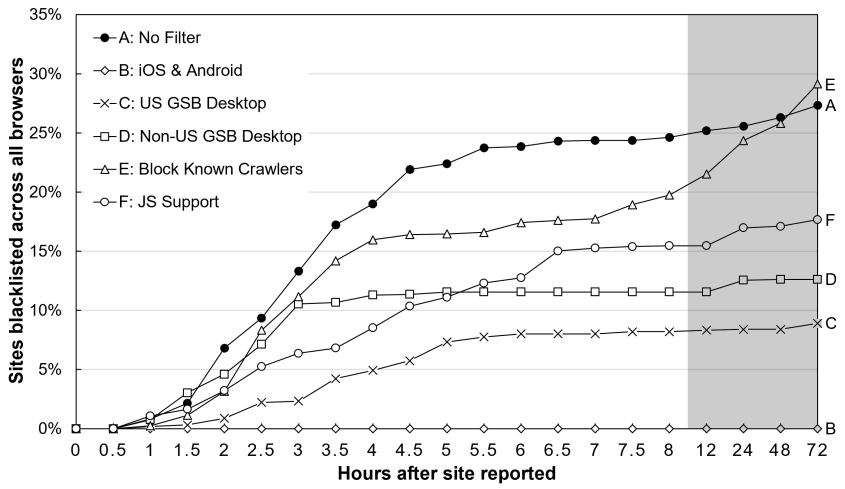



Figure 4.4: Effect of cloaking on blacklisting over time (full tests).

privacy and quick lookups but requires periodic refreshes of the entire blacklist [48]). Slow response speed has been a notable shortcoming of blacklists, and it appears that browser blacklist caching can still be improved from a security perspective[1].

Edge and IE 11 (both protected by SmartScreen) proved far less consistent. In the full tests, Edge displayed warnings up to two hours earlier *and* covered more websites than IE until the 24-hour mark. However, in the preliminary tests—which involved deceptive URLs detectable by heuristics—IE would often display preemptive warnings for websites that would not be blocked in Edge for several hours, if at all. These deviations were evident in the preliminary APWG, SmartScreen, and PayPal tests as per Table A.1 in Section A.

### *4.5.5 Filter Performance*

Figure 4.4 shows the percentage of my phishing websites blacklisted over the course of all the full experiments, grouped by cloaking filter type. Because this summary view masks some of the distinctive per-entity behavior I observed in Table 4.7, the entity-

[1]Delays caused by blacklist caching were subsequently addressed in Google Chrome in December 2019 through the addition of real-time URL lookups. I elaborate on this change in Section 5.10.

specific figures throughout Section 4.5.6 should be consulted as a supplement. Note that because these figures consider *all* browser-filter combinations, their Y-axes differ from Figure 4.3 and the $S_{bf}$ scores in Table 4.7. Nevertheless, they all convey the finding that there exist considerable gaps in today's blacklists.

The overall earliest blacklisting I observed occurred after approximately 40 minutes, in Chrome. Significant growth took place between 1.5 and 6.5 hours and continued at a slower rate thereafter for the full 72-hour period. Compared to websites without cloaking, my cloaking techniques showed moderate to high effectiveness throughout the life of each phishing website. *Filter B* saw no blacklisting whatsoever across any desktop or mobile browser I tested. *Filters E* and *F* proved most effective in the early hours of deployment, while the geo-specific *Filters C* and *D* saw the lowest amount of blacklisting in the long term. Between 48 and 72 hours after deployment, websites with *Filter E* overtook *Filter A* websites by a small margin; upon further analysis, I found that this was due to a high level of interest in such websites following reporting to the APWG. All other types of websites with cloaking were on average less likely to be blacklisted than websites without.

### *4.5.6 Entity Performance*

Although no single entity was able to overcome all of the cloaking techniques on its own, collectively the entities would be successful in doing so, with the exception of *Filter B* (this has since been corrected as per the discussion in Section 4.6.1).

#### Google Safe Browsing

Due to the high market share of the browsers it protects, GSB is the most impactful anti-phishing blacklist today. It commanded the highest score $S$ in both the preliminary and full tests. GSB's key strength lies in its speed and coverage: I observed that a crawler would normally visit one of my websites just seconds after I reported it. 94.7% of all the

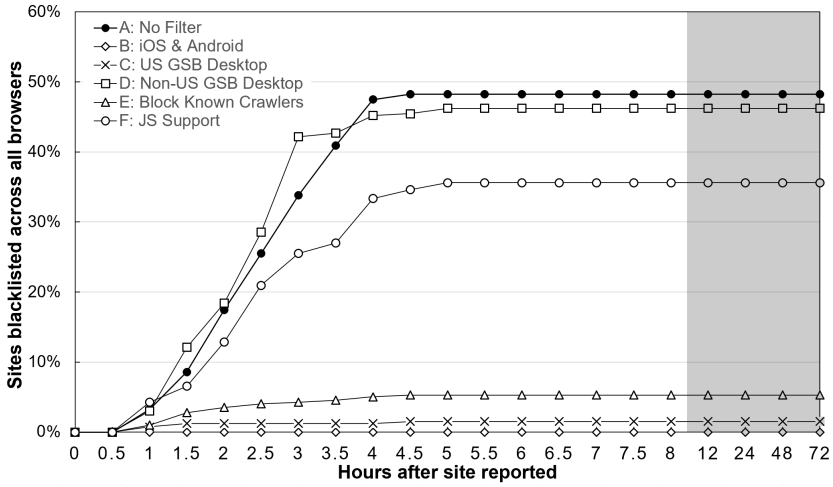



Figure 4.5: Blacklisting over time (GSB full test).

websites I reported in the full tests were in fact crawled, and 97% of websites without cloaking (*Filter A*) ended up being blacklisted. Blacklisting of *Filter D* was comparable, and *Filter F* improved over the preliminary tests.

However, GSB struggled with *Filter E*, which blocked hostnames specific to Google crawlers. It also struggled with *Filter C*, which denied non-US traffic. The reason the respective $PB_f$ scores are low is that if the initial crawler hit on the phishing website failed, GSB would abandon further crawling attempts; the initial hit almost always originated from a predictable non-US IP address. Another weakness appears to be data sharing, as none of the websites I reported to GSB ended up being blacklisted in Edge, IE, or Opera.

### Microsoft SmartScreen

SmartScreen proved to be the only native anti-phishing blacklist to leverage *proactive* URL heuristics to blacklist phishing websites, which allowed Microsoft browsers to achieve high scores during the preliminary tests. These heuristics were mainly triggered by URLs with a deceptive domain name. In the preliminary tests, Edge proved to be exceptionally well-protected, achieving a top $S_b$ score of 0.87—the highest of any browser.

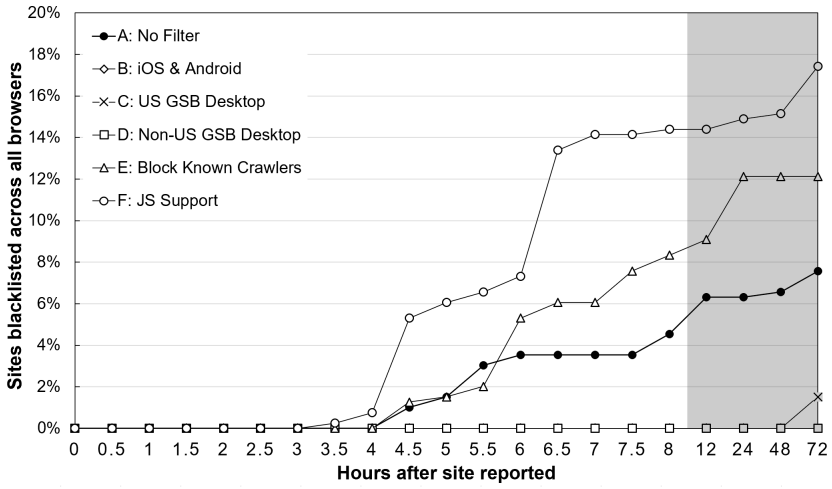



Figure 4.6: Blacklisting over time (SmartScreen full test).

In the full tests, the performance of IE improved over the preliminary tests and became more consistent with that of Edge. Surprisingly, SmartScreen was more likely to blacklist websites *with* cloaking than those without, possibly due to the use of cloaking detection or classification techniques (which would be commendable) alongside low trust of my phishing reports (see the limitations in Section 4.7).

Reporting to SmartScreen did not seem to significantly affect any non-Microsoft browsers; the entity thus shares a similar shortcoming with GSB. SmartScreen was also among the slowest entities to reactively respond to phishing reports, and its overall coverage was poor, which is its key weaknesses.

### APWG

The APWG was the second-highest scoring entity in the full tests and showed consistent protection of all browsers. Its score $S$ increased substantially compared to the preliminary tests due to improvements to bypass *Filters C* and *E*, which allowed APWG reports to result in blacklisting of such websites in GSB browsers—something not achieved when I

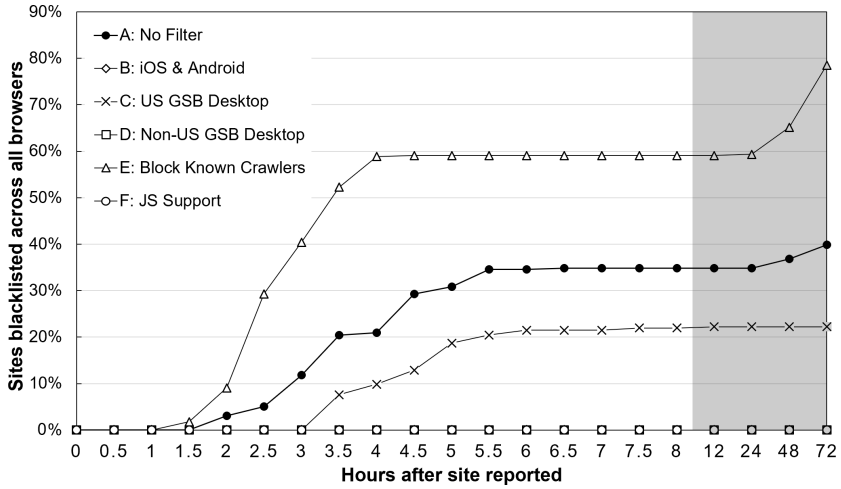



Figure 4.7: Blacklisting over time (APWG full test).

reported directly to GSB. The APWG also generated the highest level of crawler traffic of any entity I tested.

Unfortunately, the APWG failed to blacklist any websites with *Filter D* or *F* in the full tests; its preliminary success proved to have been related solely to the detection of deceptive URLs by IE and Edge. Interestingly, I saw a large increase in the blacklisting of websites with *Filter E* after the 24-hour mark; after analyzing the traffic logs I believe that this is due to data sharing with PayPal (this trend is also reflected in Figure 4.9).

### PhishTank

PhishTank is a community-driven clearinghouse that allows human volunteers to identify phishing content [29]; it also leverages a network of crawlers and partners to aid in this effort. It was the second-highest performer in the preliminary tests thanks to its expeditious blacklisting in GSB browsers. In the full tests, I was surprised to see that only 46.7% of websites reported were crawled, and very few websites were ultimately blacklisted. Despite this, PhishTank generated the second-highest volume of total crawler traffic. I do

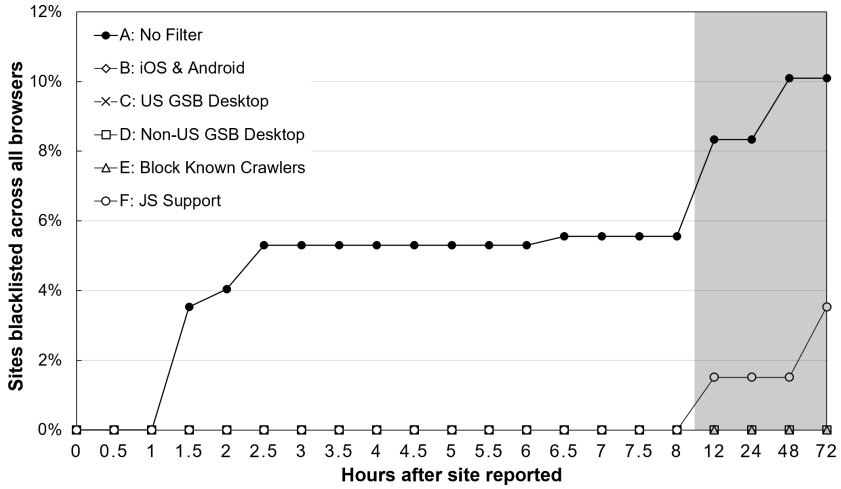



Figure 4.8: Blacklisting over time (PhishTank full test).

not know the reasons for its shortcomings and PhishTank did not reply to my disclosure; I suspect that the manual nature of classification by PhishTank may be a limiting factor.

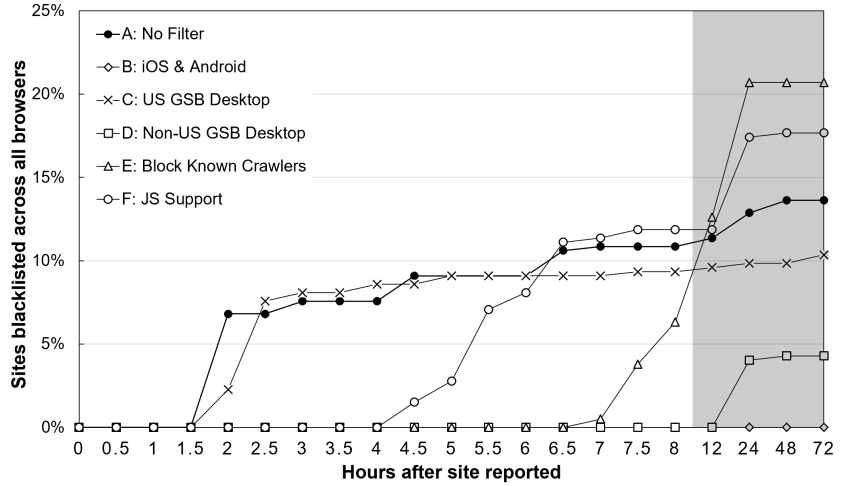



Figure 4.9: Blacklisting over time (PayPal full test).

### PayPal

During the preliminary tests, PayPal's own abuse reporting service struggled to bypass *Filters D* and *F*, but overcame the latter in the full experiments while maintaining a mod-

erate degree of blacklisting. Its protection of Opera also improved between the two tests. Despite crawling all but two of the websites I reported in the full tests, the average response time and browser protection ended up being poor overall. I cannot disclose the reasons for this but expect to see a future improvement as a result of these findings.

#### Remaining Entities

Performance scores for entities only included in the preliminary tests are found in Table A.1 within Section A. High-level descriptions of each follow.

All websites I reported to *ESET* ended up being crawled, but only a fraction of those—all with *Filter A*—were actually blacklisted. Overall, speed was slow, though the blacklisting did span multiple browsers. *Netcraft* yielded the best protection of Opera in the preliminary tests, but overall it struggled with most of the cloaking techniques and did not deliver timely blacklisting. In retrospect, given the unexpectedly poor performance of *PhishTank* in the full tests, I would have been interested in re-testing *Netcraft*. Reports to *US CERT* led to minimal crawler activity and blacklisting; disregarding heuristics from the Microsoft browsers, only a single website was blacklisted. Phishing reports I sent to *McAfee* effectively bypassed *Filter A* and *Filter E* but only appeared to lead to timely blacklisting in Microsoft Edge. Reports to *WebSense* had no effect beyond crawler traffic related to the URL heuristics used by Microsoft; while I was hopeful the e-mail reporting channel I used would prove fruitful, this is understandable given that the company focuses on the protection of its enterprise customers.

### *4.5.7 Abuse Reports*

During the preliminary testing period, my web host (Digital Ocean) received a total of only 30 abuse reports, or roughly one for every 5.5 phishing websites blacklisted. The total number of reports increased to 266 during the full tests: also a substantial relative increase

of one report per 1.7 blacklisted websites. Because I had advised the host of this research prior to deployment, the infrastructure remained online. Normally, per hosting terms of service, unresolved abuse should result in take-down (a notable exception includes "bulletproof" service providers which indirectly sponsor cybercrime [118]).

Each abuse report specified the offending URL, the nature of the abusive content (i.e., phishing), the reporting entity, the date, and the IP address of the associated web server. Due to unexpected changes to the web host's data retention policy, data for all but the final full experiment (i.e., PayPal) was limited to the latter two parameters. This limitation precludes a conclusive evaluation of the effect of cloaking on abuse reports, though I believe the increase in the raw number of reports between the preliminary tests and full tests is potentially indicative of improvements within the ecosystem.

During the full PayPal test, 52 phishing websites hosted on 30 (out of the 40) server IP addresses ended up being blacklisted. The hosting provider sent 73 abuse reports corresponding to 28 of those IPs and one IP which was not hosting any blacklisted websites. The abuse reports originated from *MarkMonitor* (43), *Netcraft* (14), *UK NCSC* (13), and *IsItPhishing.org* (3) and favored IP addresses within the EU. Of the 73 websites with abuse reports, 18 used *Filter A*, 16 used *Filter C*, five used *Filter D*, 26 used *Filter E*, and eight used *Filter F*; none used *Filter B*. The reports arrived at a median of 17 hours after each website was reported, distributed as shown in Figure 4.10. During the preliminary PayPal test, there were only six abuse reports for six IP addresses.

The relatively long delay observed before the arrival of abuse reports underscores the importance of timely blacklisting as the first line of defense against phishing attacks. Take-down of these websites would not be possible until some time after the abuse report (i.e., to give the website owner a reasonable opportunity to take action), and in some cases (e.g., the re-compromise of a legitimate server) it may even prove less effective than blacklisting.

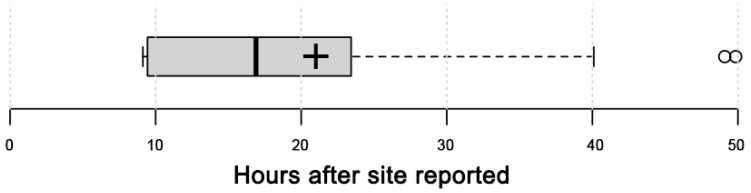



Figure 4.10: Timing of abuse reports for the full PayPal test.

### *4.5.8 Crawler Traffic Analysis*

My 2,380 phishing websites logged a total of 2,048,606 HTTP requests originating from 100,959 distinct IP addresses. A substantial proportion of requests was characterized by crawlers scanning for phishing kit archives (i.e., zip files) or credential dump files; such requests resulted in 404 "not found" errors. It is beneficial for the security community to be able to identify compromised user information and study phishing kits [56, 102], but such crawling is noisy. By monitoring traffic to their phishing websites, cybercriminals could become aware of the ecosystem's knowledge of their tools, and adapt accordingly.

Figure 4.11 aggregates the web traffic to all PhishFarm phishing websites from the full tests over a two-week period relative to initial deployment (along a logarithmic scale). Not unexpectedly, I observed the largest amount of traffic in the hours immediately after reporting each phishing website. Smaller spikes occurred several hours or days thereafter as additional infrastructure started accessing the websites. I automatically disabled each website at the end of its 72-hour deployment; crawlers would thus start seeing 404 errors thereafter. I observed a spike in traffic at this point with characteristics similar to the initial crawler traffic, followed by an immediate sharp decline (presumably once the offline state of each website was verified). Over the following seven days, I saw a consistent yet slowly-declining level of traffic. It is clear that an effort is being made to ensure that offline phishing content does not make a return. After about ten days, I noted a second sharp decline, after which the traffic reached insignificant levels. I did not study the long-

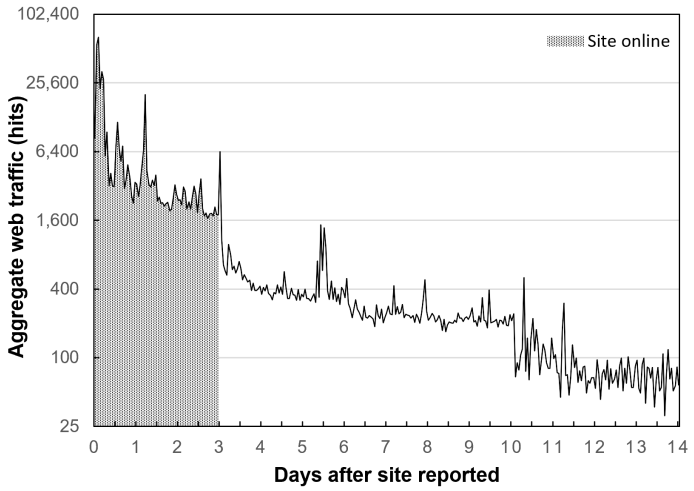



Figure 4.11: Traffic to my phishing websites over time (full tests, logarithmic scale).

term persistence of blacklist warnings within this dataset, but this is something that I do evaluate in Section 5.8.2 in the following chapter.

### Geographical Origin

Using the GeoLite2 IP database [79], I found that traffic to my phishing websites originated from 113 different countries across the majority of North America, Europe, and Asia, and some of Africa, as shown in Table 4.8. 79.02% of all unique IP addresses were based in the US; this accounted for a slightly lower 64.73% of all traffic but still constituted an overwhelming majority overall.

### Entity Crawler Overlap

I provide a summary of IP address overlap between entities in Table 4.9. The data is indicative of collaboration between certain entities, as discussed in Section 4.6.2. Although I did not perform a per-entity analysis of long-term crawler traffic, such an evaluation could be a valuable future use case of the PhishFarm framework, as it might reveal shared infrastructure or crawling patterns.

Table 4.8: Geographical distribution of crawler requests to my websites.

| Country | Total Traffic | Unique IPs |
|---|---|---|
| United States | 64.73% | 79.02% |
| United Kingdom | 6.66% | 2.42% |
| Germany | 4.72% | 1.30% |
| Brazil | 1.99% | 2.04% |
| Italy | 1.80% | 0.34% |
| Japan | 1.73% | 0.43% |
| Netherlands | 1.73% | 0.76% |
| India | 1.54% | 0.76% |
| Canada | 1.36% | 1.28% |
| France | 1.21% | 0.68% |
| Belgium | 0.85% | 0.13% |
| Singapore | 0.65% | 0.30% |
| Ireland | 0.65% | 0.66% |
| Norway | 0.65% | 0.18% |
| Australia | 0.63% | 0.34% |
| Korea | 0.50% | 0.17% |
| Denmark | 0.50% | 0.12% |
| Estonia | 0.48% | 0.07% |
| Austria | 0.45% | 0.15% |
| Russia | 0.42% | 2.23% |
| Unknown | 3.99% | 1.96% |
| 93 Others | 2.75% | 4.65% |

| *Entity* | *Unique IPs* | *IP Overlap* | | | | |
|---|---|---|---|---|---|---|
| | | **GSB** | **MS** | **APWG** | **Phish-Tank** | **PayPal** |
| **GSB** | 1,788 | | 7.94% | 31.20% | 18.40% | 53.52% |
| **MS** | 475 | 29.89% | | 29.89% | 23.16% | 38.11% |
| **APWG** | 6,165 | 11.08% | 2.30% | | 11.13% | 47.96% |
| **PhishTank** | 2,409 | 13.66% | 4.57% | 28.48% | | 47.11% |
| **PayPal** | 17,708 | 5.40% | 1.02% | 16.70% | 6.41% | |

Table 4.9: Crawler IP overlap between entities.

| | | **Total HTTP Requests** | | **Unique IP Addresses** | |
|---|---|---|---|---|---|
| | | *Valid URL* | *Invalid URL* | *Valid URL* | *Invalid URL* |
| **Prelim.** | *Sites Live* | 271,943 | 452,049 | 6,528 | 11,869 |
| | *Day 3-14* | 262,141 | | 7,230 | |
| | *Total* | 986,133 | | 20,874 | |
| **Full** | *Sites Live* | 355,093 | 545,704 | 22,929 | 54,392 |
| | *Day 3-14* | 161,676 | | 21,991 | |
| | *Total* | 1,062,473 | | 80,085 | |

Table 4.10: Web traffic during and after website deployment.

## 4.6 Security Recommendations

Based on an analysis of my experimental findings, I propose several possible improvements to the anti-phishing ecosystem.

### *4.6.1 Cloaking*

My first set of recommendations focuses specifically on the cloaking techniques tested by *PhishFarm*: In this section, I will explore the implications of potential ecosystem vulnerabilities and ways to address these vulnerabilities.

### Mobile Users

I believe that the highest priority within the current ecosystem should be effective phishing protection for mobile users. Such users are not only inherently more vulnerable to phishing attacks [131], but now also comprise the majority of web traffic [120].

My work has been impactful in better securing mobile users by enhancing existing anti-phishing systems. For over a year between mid-2017 and late 2018, GSB blacklists (with nearly a 76% global market share at the time) simply did not function properly on mobile devices: *none* of my phishing websites with *Filter A*, *E* or *F* (targeting both desktop and mobile devices) showed any warnings in mobile Chrome, Safari, or Firefox despite being blacklisted on desktop. I confirmed the disparity between desktop and mobile protection through periodic small-scale testing, and by analyzing a separate dataset of traffic to real phishing websites, as discussed in Chapter 6. Following my disclosure, I learned that the inconsistency in mobile GSB blacklisting was due to the transition to a new mobile API designed to optimize data usage, which ultimately did not function as intended. Because blacklisting was not rectified after the full tests, I contacted the entities anew. As a result, in mid-September 2018 Mozilla patched Firefox (from version 63) such that all desktop warnings were also shown on mobile. Google followed suit days thereafter with a GSB API fix that covered mobile GSB browsers retroactively; mobile Chrome and Safari now mirror desktop listings, albeit with a shorter-lived duration to lower bandwidth usage. Chrome, Safari, and Firefox thus again join Opera as mobile browsers with effective blacklists, though some popular mobile browsers still lack such protection [131].

Upon close inspection of the preliminary test results, I found that *Filter B* websites were solely blacklisted due to their suspicious URLs rather than my reports. During the full tests, not a single website with *Filter B* was blacklisted in any browser, interestingly despite the fact that crawlers did successfully retrieve many of these websites. GSB ad-

dressed this vulnerability in mid-September 2018, together with the aforementioned API fix. Through a subsequent final redeployment of PhishFarm, I verified that websites with *Filter B* were being blacklisted following reports to GSB, the APWG, and PayPal. Other entities—including ones I did not test—should ensure that websites targeted at mobile users are being effectively detected.

#### Geolocation

Although my experiments only considered two simple geolocation filters (US and non-US), my findings are indicative of exploitable weaknesses in this area. Given the overwhelming amount of crawler traffic from the US (79%, per Table 4.8), *Filter C* should *not* have been as effective as it proved to be. I hypothesize that other geo-specific filters would have similarly low blacklisting rates, in part due to the crawler characteristics discussed in Section 4.5.6. Country- or region-specific filtering paired with localized page content is not an unusual sight in real-world PayPal phishing kits that I have analyzed.

#### JavaScript

It is trivial to implement JavaScript-based cloaking such as *Filter F*. This technique proved to be effective in slowing blacklisting by three of the five entities in the full tests. Fortunately, SmartScreen bypasses this technique well, and PayPal started doing so following my disclosure. The broader ecosystem should better adapt to client-side cloaking, especially if the sophistication of such cloaking increases over time.

### *4.6.2 Anti-phishing Entities*

I also offer a number of more general recommendations for anti-phishing systems of tomorrow.

### Continuous Testing

The mere failure of mobile blacklisting that I observed during our experiments is sufficient to warrant the need for continuous testing and validation of blacklist performance. Periodic deployment of PhishFarm could be used for such validation. In addition, continuous testing could help ensure that future cloaking techniques—which may grow in sophistication—can effectively be mitigated without compromising existing defenses. As an added benefit, trends in the relative performance of different entities and the overall speed and coverage of blacklisting could be modeled.

### Trusted Reporting Channels

The phishing reporting channels I tested merely capture a suspected URL or a malicious e-mail. While the latter is useful in identifying spam origin, I believe a better solution would be the implementation of standardized trusted phishing reporting systems that allow the submission of specific metadata (such as victim geolocation or device). Trusted channels could allow detection efforts to more precisely target high-probability threats while minimizing abuse from deliberate false-negative submissions; they could also simplify and expedite collaboration efforts between anti-phishing entities and abused brands, which may hold valuable intelligence about imminent threats.

### Blacklist Speed

The gap between the deployment of a phishing website and its blacklisting across browsers represents the prime window for phishers to successfully carry out their attacks. At the level of an individual entity, cloaking has a stronger effect on the *occurrence* rather than the speed of blacklisting. However, if I look at the ecosystem as a whole in Figure 4.4, cloaking clearly delays blacklisting overall. My test results show that blacklisting now typically occurs in a matter of hours—a stark improvement over the day- or week-long

response time observed years ago [92, 114]. However, given the tremendous increase in total phishing attacks since then (on the order of well over 100 attacks per hour in 2018 [5]), I believe that even today's 40-minute best-case blacklist response time is too slow to deter phishers and effectively protect users. The gap needs to be narrowed, especially by slower entities (i.e., those not directly in control of blacklists). Future work should investigate the real-world impact of delays in blacklisting on users and organizations victimized by phishing attacks in order to accurately establish an appropriate response threshold.

### Phishing Volume

GSB, the APWG, and PayPal crawled nearly all of the phishing websites I reported. In particular, GSB proved to deliver a consistently agile response time despite the high number of reports I submitted. Other entities fell short of this level of performance. With the increasing volume of phishing attacks being observed in the wild, it is essential that all players in the ecosystem remain robust and capable of delivering a consistent response.

### Data Sharing

Data sharing has long been a concern within the ecosystem [90]. I found that the two main blacklist operators (GSB and SmartScreen) did not appear to effectively share data with each other, as per Table 4.7. However, clearinghouse entities (APWG and PhishTank) and PayPal itself showed consistent protection across all browsers. Unfortunately, the speed and overall coverage of clearinghouses appear to be inferior to those of the blacklist operators in their respective browsers. Closer cooperation could thus not only speed up blacklisting, but also ensure that malicious websites are blocked universally. Strengthening this argument, perhaps a breakdown in communication between the systems used by different entities accounted for the phishing websites that were successfully crawled, but not ultimately blacklisted during my experiments.

## 4.7 Limitations

My findings are based on a controlled (to the extent possible without sending out real spam e-mails) empirical experiment and observations from a large set of supporting metadata and a high volume of anti-abuse crawler traffic. My study focuses exclusively on *native* phishing blacklist protection that is available by default in the browsers and platforms tested. Systems with third-party security software may see enhanced protection [114], though cloaking can also have an impact on the systems powering such software.

Within its scope, my analysis should still be considered with certain limitations in mind. I suspect that real-world blacklisting of phishing attacks may be more timely than what my results otherwise suggest, as my efforts to isolate cloaking as a variable in the experimental design (i.e., by using randomized domain names, never rendering the actual phishing content in browsers being monitored, and using *.com* domains months after registration) also eliminated many of the methods that the ecosystem can use to detect or classify phishing (e.g., URL-, network-, or DNS-based analysis). However, this reduced detectability is offset, to an extent, by the possibility of malicious actors to likewise evade such forms of detection. I observed in the preliminary tests that only URLs containing brand names were quicker to be blacklisted than others; in the wild, there is also a shifting tendency to abuse compromised infrastructure and distribute random phishing URLs in lieu of more deceptive alternatives [41, 102]. In terms of factors under my control, it was not financially feasible to achieve a one-to-one mapping between IP addresses and all of my domains; this is a skewing factor which may have acted in favor of blacklists in the full tests, such as with the ten *Filter D* websites which were blacklisted despite not being successfully retrieved during the GSB experiment.

Finally, I only submitted a *single* and *direct* report for each phishing website deployed. Although real-world phishing websites might see a much higher volume of automated

reports (e.g., from sources such as spam filters), the volume of per-URL phishing reports in the wild can, in fact, be reduced by attackers (e.g., through the use of redirection links). More importantly, direct reports such as ours (in particular to the blacklist operators) might be subject to more suspicion because anti-phishing entities must account for adversaries who willingly submit false reports or seek to profile crawling infrastructure. Although the blacklist operators to which I disclosed did not express concern with my reporting methodology, I learned that the crawling infrastructure used to respond to direct reports is indeed designed to be unpredictable to mitigate adversarial submissions. SmartScreen's lower crawl rate may be explained by this; GSB, on the other hand, consistently responded quickly and seemed to give the direct reports a high level of priority. It is, therefore, possible that either the classification of reporting channel abuse works very accurately, or that reporting channels are more vulnerable to adversarial submissions than what the entities otherwise believe; regardless, to improve the future effectiveness of reporting, I propose an alternative approach in Section 4.6.2 and I implement this approach in Chapter 5.

Ultimately, if each report represents a chance that a phishing website will be blacklisted, I believe that my experimental design still captures trends therein; moreover, my findings with respect to cloaking effectiveness are consistent with internal PayPal e-mail and web traffic data pertaining to actual victims of phishing. To address its current limitations, the PhishFarm framework could be adapted to follow a different (possibly collaboratively arranged) reporting methodology, consider a broader range of cloaking techniques, or even be applied to proactively-detected live phishing URLs for which cloaking can be profiled.

### Statistical Tests

In the full tests, I was surprised to find the blacklisting performance of each entity with respect to the different filters to have far more clear-cut gaps than in the preliminary tests; 11 of the 30 the per-entity filter groups saw no blacklisting whatsoever (per Table 4.7). Although ANOVA as originally planned could still be meaningfully applied to the subset of entities which had three or more filter groups that satisfied homogeneity of variance [20] (i.e., had some blacklisting activity), I chose not to perform such tests as the resulting power would be below my target, and instead relied on direct observations supported by crawler metadata. If my experiments were to be repeated to validate improvements in blacklisting or continuously test the ecosystem, I believe that statistical tests could be highly useful in assessing whether per-entity blacklisting performance improved significantly for each respective cloaking filter.

## 4.8 Related Work

To the best of my knowledge, the work in this chapter is the first controlled effort to measure the effects of cloaking in the context of phishing. Several prior studies measured the general effectiveness of anti-phishing blacklists and the behavior of phishing kits; none of the prior work I identified considered cloaking, which may have had a skewing effect on the datasets and ecosystem findings previously reported. Cloaking itself has previously been studied with respect to malicious search engine results: Invernizzi et al. [58] proposed a system to detect such cloaking with high accuracy. I later studied the nature of server-side cloaking techniques within phishing kits and proposed approaches for defeating each (as discussed in Chapter 3) [102].

The work most similar to PhishFarm is NSS Labs' [99] recent use of a proprietary distributed testbed [98] to study the speed of native phishing blacklisting in Chrome,

Firefox, and Edge. The main limitation of NSS Labs' approach is the reliance on feeds of known phishing attacks; any delay in the appearance of a website in each source feed can affect the accuracy of blacklisting time measurements. Furthermore, phishing websites could be overlooked in the case of successful cloaking against the feeds. I address these limitations by having full control over the deployment and reporting time of phishing websites.

Sheng et al. [114] took an empirical approach to measure the effectiveness of eight anti-phishing toolbars and browsers powered by five anti-phishing blacklists. The authors found that heuristics by Microsoft and Symantec proved effective in offering zero-hour protection against a small fraction of phishing attacks, and that full propagation across phishing blacklists spanned several hours. This work also found that false-positive rates in blacklists are near-zero; I thus did not pursue such tests in my experiments. While Sheng et al.'s work was based on phishing websites only 30 minutes old and was thus better controlled than earlier blacklist tests [72], the datasets studied were of limited size and heterogeneous in terms of victim brands; the anti-phishing tools evaluated are now dated. In addition, Sheng et al. checked blacklist status with a granularity of one hour—longer than PhishFarm's ten minutes.

Han et al. [51] analyzed the lifecycle of phishing websites by monitoring cybercriminals' behavior on a honeypot web server. The authors timed the blacklisting of the uploaded websites across Google Safe Browsing and PhishTank. Uniquely, this work sheds light on the time between the creation and deployment of real phishing websites. A key difference in this work is my ability to customize and test different configurations of phishing kits instead of waiting for one to be uploaded. For instance, I could target specific brands or configure my own cloaking techniques to directly observe the ecosystem's response. My experiments suggest a significantly faster blacklist response time by

the ecosystem than what Han et al. found; my test sample size was also nearly five times larger.

Virvilis et al. [131] carried out an evaluation of mobile web browser phishing protection in 2014 and found that major mobile web browsers at the time included no phishing protection. Like that of Sheng et al., this evaluation was empirical and based on a set of known phishing URLs. In my work, I found that mobile Chrome, Safari, and Firefox now natively offer blacklist protection, but that this protection was not functioning as advertised during my tests.

The differences between today's phishing trends and those seen in prior work show that the ecosystem is evolving quickly. This warrants regular testing of defenses and re-evaluation of criminals' circumvention techniques and attack vectors; it also underscores the importance of scalable and automatable solutions. My testbed shares some similarities with previous work [99, 114] but it is the only such testbed to offer full automation without the need for intervention during the execution of experiments, and the only one to actively deploy websites and directly send reports to entities. It thus lends itself well to a longitudinal measurement of anti-phishing blacklists against the latest threats, which is essential to gaining a practical understanding of their efficacy [75].

## 4.9 Summary

By launching and monitoring a large set of phishing websites, I carried out the first controlled evaluation of how cloaking techniques can hamper the speed and occurrence of phishing blacklist warnings in modern web browsers. As a result of my disclosure to anti-phishing entities, mobile blacklisting is now more consistent, and some of the cloaking techniques I tested are no longer as effective; others represent ongoing vulnerabilities that could be addressed through tweaks to existing detection systems. Such tweaks should also seek to improve the overall speed of blacklists to better counter the modern on-

slaught of phishing attacks. Blacklist protection should not be taken for granted; continuous testing—such as that supported by my framework—is key to ensuring that browsers are secured sufficiently, and as intended. In the next chapter, I propose methodology for carrying out such testing while simulating the ever-evolving state of the ecosystem.

Cloaking carries a detrimental effect on the occurrence of blacklisting due to the fundamentally-reactive nature of the main detection approach currently used by blacklists; it thus has the potential to cause continuous damage to the organizations and users targeted by phishers. Although no single entity I tested was able to individually defeat all of my cloaking techniques, collectively the requisite infrastructure is already in place. In the short term, collaboration among existing entities could help address their individual shortcomings. I observed that proactive defenses (such as the URL heuristics of SmartScreen) proved to deliver superior protection—but only under the right circumstances. In the long term, the ecosystem should move to more broadly implement general-purpose proactive countermeasures to more reliably negate cloaking. It is important for the ecosystem to be able to effectively bypass cloaking because it is merely one way in which phishing websites can be evasive. For instance, with cloaking alongside redirection chains or bulletproof hosting, phishing websites might otherwise avoid existing mitigations far more successfully than what I have observed.

Phishing has proven to be a difficult problem to solve due to attackers' unyielding persistence, the cross-organizational nature of infrastructure abused to facilitate phishing, and the reality that technical controls cannot always compensate for the human weakness exploited by social engineers. I believe that continuous and close collaboration between all anti-abuse entities, which can lead to a deep understanding of current threats and development of intelligent defenses, is the crux of optimizing controls and delivering the best possible long-term protection for phishing victims.

Chapter 5

# PHISHTIME: CONTINUOUS LONGITUDINAL MEASUREMENT OF THE EFFECTIVENESS OF ANTI-PHISHING BLACKLISTS

## 5.1 Introduction

Due to their ubiquity in modern web browsers, anti-phishing blacklists are a key defense against phishing attacks, and they have become particularly important amid the large volume of phishing websites that currently plague the Internet. Meanwhile, phishers are engaged in a cat-and-mouse game with the anti-phishing ecosystem: sophistication in phishing websites—such as the use of evasion techniques—continues to grow. Yet, the effectiveness of blacklisting of websites with such evasion techniques is difficult to measure, and there have been no methodical efforts to make and track such measurements, at the ecosystem level, over an extended period of time.

Despite the importance of blacklists, and even attention from security researchers [101, 111, 114, 131], there have been no systematic, real-world, *long-term* studies of the performance of anti-phishing blacklists. *Evasive* phishing attacks that seek to circumvent blacklists are not only becoming more common, but have been shown to be responsible for the majority of the real-world impact caused by traditional phishing [103]. Thus, the effectiveness of the blacklisting of such attacks warrants scrutiny. In the previous chapter, I showed that various *cloaking* techniques effectively slow or prevent the blacklisting response of several key anti-phishing entities. Similar methodology can be adapted to evaluate not only individual entities, but the collective response of the ecosystem, which can provide deeper insight into the level of protection provided to users.

In this chapter, I leverage and enhance the *PhishFarm* framework (described in Section 4.4) to develop a new system, *PhishTime*, for continuously and automatically detecting unmitigated phishing websites that are online, for replicating key aspects of their configuration in a controlled setting, and for generating longitudinal experiments to measure the consistency of the ecosystem's defenses against them. These experiments focus on evaluating the performance (*speed* and *coverage*[1]) of anti-phishing blacklists, with the goal of improving it. By continuously identifying sophisticated phishing attacks in the wild *and* by continuously monitoring the response of anti-phishing blacklists to these sites' evasion techniques, with minimal manual effort, PhishTime can identify gaps within blacklisting as well as gaps in the broader anti-phishing ecosystem.

In the first longitudinal study of its kind, I deploy the PhishTime framework over the course of one year to measure the performance of three blacklists—*Google Safe Browsing*, *Microsoft SmartScreen*, and *Opera*—across major desktop and mobile browsers (which collectively have an overwhelming global market share [95]). During this period, I systematically launched and reported 2,862 new and previously unseen (innocuous) PayPal-branded phishing websites as part of six large experimental deployments. I designed my experiments using observations from real phishing websites detected with the help of my framework. As a result, I configure my own websites with a variety of evasion techniques representative of phishing attacks in the wild. I then precisely evaluate blacklists' response to these websites. Similarly to the original PhishFarm experiments, these experiments were carefully controlled to avoid confounding effects (e.g., each website used a unique, previously-unseen *.com* domain).

I find that although blacklists have an average response time of as little as 55 minutes against unsophisticated phishing websites, phishing websites with evasion techniques commonly used in the wild, such as redirection links with server-side cloaking, delay

[1]These metrics are discussed in more detail in Section 5.3

blacklists' response up to an average of 2 hours and 58 minutes, and are up to 19% less likely to be detected. I also find it feasible for attackers to re-use domains for multiple phishing websites: with evasion, such websites are still blacklisted up to 1 hour and 20 minutes slower than unevasive ones. Even such a seemingly short delay can cause 20% more users to fall victim to each attack [103]. Therefore, delays to blacklisting as a result of evasion represent a serious security concern.

As part of my tests, I show that new *evidence-based* phishing reporting protocols (i.e., that allow the submission of evidence such as a screenshot in addition to a URL) [23] are essential to improving the protection offered by blacklists, and I perform the first comparison of such a protocol alongside traditional URL reporting [46]. In addition, I confirm the improvements made by Mozilla to mobile Firefox as a result of the PhishFarm disclosure (discussed in Section 4.6.1), but I also find that mobile Chrome and mobile Safari continue to receive a far lesser degree of protection than their desktop counterparts. Lastly, I identify several other weaknesses within the ecosystem, including a class of *behavior-based* JavaScript evasion techniques (discovered by PhishTime) that completely avoided blacklisting during my experiments.

I conclude that enhanced anti-phishing protection on mobile devices, and the expansion of evidence-based phishing reporting protocols, are critical ecosystem improvements that could better protect users against modern phishing attacks, which are marked by a high degree of evasiveness against detection infrastructure. I disclosed my findings to the corresponding entities within the ecosystem, and I am collaborating with the Anti-Phishing Working Group (APWG) to permanently integrate the PhishTime framework as an ecosystem service. The contributions of this chapter are thus as follows:

- A highly automated framework for the continuous long-term empirical measurement of the anti-phishing ecosystem.

- Deployment of the framework for a longitudinal evaluation of the performance of browser blacklists, with a focus on evasive phishing.
- An evaluation of enhanced *evidence-based* phishing reporting protocols.
- Identification and disclosure of ecosystem vulnerabilities exploitable by phishers.

## 5.2 Background

As we observed in Chapter 4, browser blacklists are a key anti-phishing defense that protects users transparently and is enabled by default in most major web browsers across both desktop and mobile devices [101]. Thus, blacklists are capable of protecting users on the same scale at which phishing occurs.

### *5.2.1 Blacklist Evasion Techniques*

A notable weakness of blacklists is that they are inherently *reactive*. Phishers capitalize on the time gap between a phishing website's deployment and its subsequent blacklisting, and may increase return-on-investment (ROI) by prolonging this gap [51, 94]. Because blacklist detection relies on content verification, blacklists are vulnerable to evasion techniques which, when successful, may *delay* or *entirely prevent* blacklisting [101]. In Section 5.4, I describe my approach to testing evasion techniques commonly used in the wild.

**Cloaking** is an evasion technique that seeks to hide phishing content from blacklist infrastructure (i.e., web crawlers) while keeping it visible to human victims [58]. As we already discussed, when a phishing website with cloaking suspects that a request is from a crawler, it will replace the phishing content with a benign-looking page or an error message. Cloaking has become standard in phishing kits and is commonly implemented on both the server-side and client-side, by applying filters based on HTTP request attributes and device characteristics [102].

**Redirection links** make it more difficult for anti-phishing systems (e.g., e-mail spam filters or blacklists) to correlate a link in a lure with a known phishing URL [19]. Because blacklists block phishing websites based on their URLs, phishers typically distribute lures with different URLs that then redirect [40] to the final phishing URL. The HTTP redirection chain itself may implement cloaking to further evade detection, and a one-to-many mapping may exist between redirection links and phishing websites to dilute each link's perceived maliciousness [138]. Phishers commonly abuse URL shortening services to create redirection links [19].

**Compromised infrastructure** is regularly used by phishers to host phishing kits [1, 65]. Such infrastructure—which otherwise contains legitimate websites—poses a particular challenge to blacklists, as the blacklists must ensure that the legitimate content is not blocked alongside the phishing content (e.g., may only differ in the *path* of a URL on the same domain [10]). Some phishing kits exploit this phenomenon by generating many sub-folders under one domain, all of which must then be individually blacklisted [103].

### *5.2.2 Reporting Protocols*

Just as individual users rely on browser blacklists to stay safe from phishing, the organizations impersonated by phishing websites rely on blacklists to protect their customers. These organizations typically obtain phishing reports from their customers and forward the identified URLs to blacklists, either directly or through the help of third-party vendors [106].

Blacklists predominantly accept reports of phishing websites in the form of a bare URL [44, 48, 84, 97]. However, such reports can prove ineffective if the website successfully uses evasion, as the blacklist may mistake the website as benign and thus fail to act appropriately on the report. Reporting protocols that facilitate the submission of ad-

ditional evidence (e.g., screenshots or page source) are currently available on a limited scale [23]; I test one such protocol in Section 5.8.6.

## 5.3 Blacklist Evaluation Metrics

In this section, I explain the metrics that I use to evaluate blacklists in the remainder of this chapter. I also describe the specific blacklists that will be considered.

### *5.3.1 Blacklist Performance*

For the longitudinal study in this chapter, I generalize the evaluation metrics originally used by PhishFarm (in Section 4.5.2) to better capture the practical characteristics of blacklists.

*Discovery* refers to a blacklist's ability to identify new URLs in the wild that are suspected of hosting phishing content. A blacklist with ideal discovery would know of every URL within the population of live phishing URLs. Discovery can result from direct phishing reports or other ecosystem sources, such as monitoring of e-mail spam, website content, or web traffic [11, 32, 71, 100, 103].

*Detection* refers to a blacklist's ability to correctly classify the *discovered* URLs, such that URLs with phishing content are added to the blacklist. A blacklist with ideal detection would not only flag every true-positive phishing URL, but it would do so promptly at the time of discovery to minimize the potential damage caused by the phishing website. Thus, I can break detection into two sub-metrics: For any set of phishing URLs discovered by a blacklist, *coverage* is the proportion of these URLs which are blacklisted at any point while they host live phishing content. *Speed* is the amount of time that elapses between

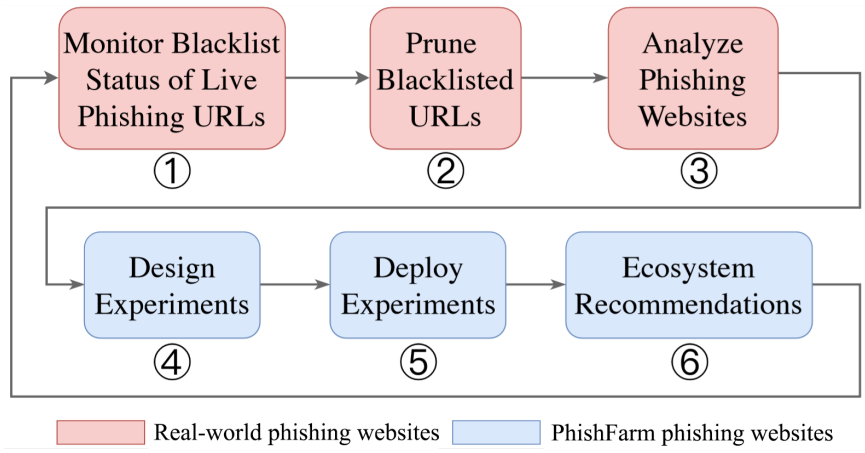



Figure 5.1: High-level overview of the PhishTime framework.

discovery and blacklisting. It is therefore desirable for blacklists to deliver high coverage and fast speed[2].

### 5.3.2 *Selection of Blacklists*

Throughout this chapter, as in Chapter 4, I consider three major browser blacklists: *Google Safe Browsing* (GSB), *Microsoft SmartScreen*, and *Opera's fraud and malware protection*. I focus my evaluation on these blacklists because of their large global market share [95]. I pay particular attention to GSB due to its overwhelming potential impact.

My methodology and framework could naturally be applied to evaluate any other blacklists or browser-based detection systems.

## 5.4 PhishTime Overview

As I discussed in the previous chapter, an effective way to empirically evaluate the performance of anti-phishing blacklists is to deploy a large *batch* of specially-configured test phishing websites, report the websites directly to blacklists, and then monitor each

[2]Perfect detection is nontrivial in part because blacklists must maintain a very low false-positive rate to avoid disrupting legitimate websites [136].

website to see if and when it is blacklisted [101, 106]. Thus, for my longitudinal evaluation of blacklist performance, I make a series of such deployments, at regular intervals, over an extended period of time. Within each deployment, I configure multiple distinct batches of websites to support different experiments.

The goal of my experiments is to provide insight into potential gaps within the ecosystem, which could, in turn, lead to actionable security recommendations. Therefore, I seek to closely replicate the phishing website configurations (i.e., evasion techniques) used by attackers. To identify such configurations and guide my experimental design, I develop the *PhishTime* framework, as shown in Figure 5.1.

The PhishTime framework is a systematic, semi-automated approach for identifying evasive (i.e., unmitigated) phishing websites in the wild. I use the framework to characterize both *typical* and *emerging* evasion techniques used by real phishing websites. Understanding the ecosystem's response to typical phishing enables identification of gaps currently being exploited by attackers, whereas analysis of less prominent emerging evasion techniques allows me to take a proactive approach to mitigate the expansion of sophisticated developments in phishing.

First, I continuously build a sample of real, live phishing URLs (❶). In my deployment, I collected PayPal phishing URLs listed in the APWG eCrime Exchange (a key clearinghouse of phishing URLs) [7] as well as URLs reported directly to PayPal via e-mail. I then start periodically monitoring the status of each URL on blacklists of interest (❷) for as long as it is live. If a URL is not initially blacklisted, I report it to the blacklists, and to various other anti-phishing entities, in an effort to get it blacklisted (using the reporting infrastructure and approach described in Section 5.7). As I continue monitoring blacklist status, I prune URLs that are blacklisted within a reasonably short time period and I retain those that are not (❸). I chose a blacklisting cutoff of two hours to eliminate URLs that blacklists could successfully *detect*, but likely originally failed to *discover* [101].

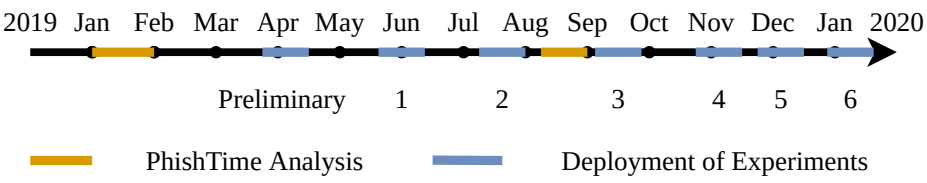



Figure 5.2: Timeline of framework & experiment deployments.

I then inspect the remaining URLs to identify why they have been evading blacklist detection. I manually analyze the evasion techniques used as well as the behavior (i.e., general appearance and user interface) of the website (❹). I performed this step by first visiting each URL, and then testing different variations of request parameters until the content could successfully be retrieved. Thus, it is possible to infer the server-side evasion techniques used by each phishing website. I also analyze each website visually, and I look at client-side source code to not only understand any evasion logic, but to compare the websites to samples of known phishing kits (made available to me) to determine which are the most common.

After analyzing a representative sample of URLs, I abstract the key trends that I observed and design experiments to replicate them (❺, Section 5.6). Finally, I deploy these experiments (❻, Section 5.7) and use the findings to make security recommendations for specific blacklists or for the ecosystem as a whole (❼, Sections 5.8 and 5.10). The findings can then influence the design of experiments in successive deployments of the framework.

## 5.5 PhishTime Analysis

I used the PhishTime framework in January 2019 to identify phishing websites in the wild capable of successfully evading blacklisting for extended periods of time. I show a timeline of my experiment deployments, and my ecosystem analysis using PhishTime, in Figure 5.2. I obtained permission from PayPal, Inc. to use PayPal-branded phishing web-

sites throughout my experiments[3]. Therefore, in my analysis, I also focus on PayPal phishing websites in the wild. I then characterized *typical* evasion techniques used by these websites, and I designed experiments which entailed deploying a series of PhishFarm-crafted phishing websites to empirically measure the response of blacklists to these techniques in a controlled manner. Later, in August 2019, I used the framework to identify less common (but more sophisticated) *emerging* evasion techniques, and I designed additional experiments to test these techniques.

### *5.5.1 Typical Evasion Techniques*

In total, I analyzed 4,393 distinct phishing URLs in the wild and I found 183 fail to be promptly blacklisted. Although this may seem like a relatively small number, prior work has shown that the majority of real-world damage from phishing occurs from a small fraction of known phishing URLs [103]. Moreover, the total URL count for the ecosystem as a whole would likely be far higher, as I only focused on a single brand in my analysis.

Of these 183 websites, 96 were never blacklisted anywhere before going offline (the average observed lifespan was 17 hours, 12 minutes), 87 were ultimately blacklisted in at least one desktop browser (with an average observed speed of 7 hours, 4 minutes) and 23 were ultimately blacklisted in at least one mobile browser (with an average observed speed of 12 hours, 2 minutes). I also observed ten websites which remained live, without blacklisting, for over one week. Note that due to the inherent delay between an attacker's deployment of a phishing URL and its appearance in a report or feed, the aforementioned timings represent lower bounds.

By analyzing URLs in the e-mail lures reported to PayPal, I found that 177 of these websites had *lure* URLs which redirected to a different final landing URL with the phishing content. I observed redirection URLs both through third-party redirection services and

[3]In the current ecosystem, PayPal is among the brands most commonly targeted by phishers [127].

attacker-controlled domains. In the latter case, I commonly observed JavaScript-based redirection alongside traditional HTTP redirection [40]. I also observed that at least 146 of these websites used some form of server-side cloaking [102]: I was unable to retrieve their content using a cloud-based web crawler but succeeded when using a mobile IP address or anonymous VPN service.

At least 42 websites had URLs with multiple paths or subdomains on the same domain, which reflects phishers' tendency to re-use infrastructure.

### *5.5.2 Emerging Evasion Techniques*

I found that eight of the 96 phishing websites which were never blacklisted implemented clever mechanisms to evade detection: five presented visitors with a CAPTCHA challenge before displaying the phishing content, two required the user to click on a button in a popup window prior to redirecting to the phishing page, and one would not render content prior to detecting mouse movement. I refer to these evasion techniques as *behavior-based*, because they require a specific user behavior to display phishing content.

## 5.6 Experimental Design

We now transition from simply observing the ecosystem to actively measuring it: to methodically test the phishing website configurations observed in the wild, I replicate them across a large sample of my own (innocuous) phishing websites which I deploy, report the respective URLs to anti-phishing entities, and monitor the speed and coverage of blacklists as they respond to my reports.

In total, I made one preliminary deployment in March 2019, and six main experimental deployments at regular intervals between May 2019 and January 2020. The purpose of the preliminary deployment—which mirrored the configuration of the first main deployment—was to verify the soundness of my experimental design and technical cor-

rectness of my framework. I summarize my deployments in Table 5.1; a more explicit breakdown is found in Table B.1 in Appendix B.

Across the main deployments, I launched 2,862 phishing websites as part of seven different experiments. I registered a total of 2,646 new *.com* domain names for these websites. Because some of my experiments involved redirection links, an additional 1,296 such links bring my total URL count to 4,158. As my experiments seek to make multiple different measurements over time, each deployment includes several different experiments.

Each experiment consists of one or more distinct *batches* of phishing websites: groups that share a single configuration corresponding to a specific *experiment*. I chose my batch size, 54, by dividing the estimated number of total batches by the number of domain names I could purchase for this research. This sample size is similar to prior empirical measurements and is designed to be large enough to support statistically-significant inferences [101].

| Experiment | | Deployment | | | | | | Per Deployment | | | Total | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | *1* *May* | *2* *Jul.* | *3* *Sep.* | *4* *Oct.* | *5* *Nov.* | *6* *Dec.* | *Batches* | *Websites* | *URLs* | *Batches* | *Websites* | *Domains Registered* |
| *A* | *Baseline* | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | 1 | 54 | 54 | 6 | 324 | 324 |
| *B* | *Basic Evasion* | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | 1 | 54 | 54 | 6 | 324 | 324 |
| *C* | *Typical Evasion: Redirection* | ✓ | ✓ | ✓ | ✓ | | | 3 | 162 | 324* | 12 | 648 | 1,080 |
| *D* | *Domain Reuse* | ✓ | ✓ | ✓ | ✓ | | | 3 | 162 | 324* | 12 | 648 | 0 |
| *E* | *Discovery* | ✓ | ✓ | ✓ | ✓ | | | 2 | 108 | 108 | 8 | 432 | 432 |
| *F* | *Emerging Evasion* | | | | | ✓ | | 7 | 378 | 378 | 7 | 378 | 378 |
| *G* | *Evidence-Based Reporting* | | | | | | ✓ | 2 | 108 | 108 | 2 | 108 | 108 |

Table 5.1: Experiments conducted during each main deployment (*half are redirection links).

### 5.6.1 Measuring Blacklist Speed and Coverage

The experiments in this section focus primarily on measuring the *detection* performance (i.e., speed and coverage) of blacklists. As I believe that it is generally infeasible for attackers to avoid *discovery* when conducting traditional phishing attacks (e.g., at scale through e-mail spam), my reporting methodology seeks to ensure that all URLs I deploy as part of these experiments are promptly discovered by the blacklists I test. I do so by simultaneously reporting the URLs to multiple blacklists and other anti-phishing entities, which I elaborate on in Section 5.7.2.

**Experiment A: Baseline.** For my simplest experiment, I launch a single batch of basic phishing websites, with no evasion technique, once in each deployment. These, and all other websites I deploy, used HTTPS to match typical phishing websites in the wild [7]. This experiment serves two key purposes: to establish a baseline for the best-case speed and coverage provided by blacklists (which can then be compared to other experiments) and to measure if these metrics remain consistent over time.

**Experiment B: Basic Evasion.** In this experiment, I test two straightforward cloaking techniques inspired by my observations in Section 5.5.1: websites that allow only traffic from browsers with a mobile *user agent* [40, 58], and websites which render content using JavaScript. I alternate these two cloaking techniques between deployments.

This experiment allows us to evaluate blacklists' response to slightly more sophisticated phishing by comparing against the baseline response in Experiment A. It also establishes a point of comparison for even more sophisticated phishing in later experiments. A secondary objective of these experiments is to assess blacklist coverage (on mobile devices) of phishing websites aimed specifically at mobile users. Mobile devices have historically been prone to phishing [131], and recent work has shown that mobile blacklists only contain a subset of known phishing URLs [101].

**Experiment C: Typical Evasion (Redirection).** Each deployment of this experiment consists of three batches of websites, with a focus on evaluating the evasiveness of redirection. In a 1-1 mapping, I pair each phishing website with a different URL (on a different domain) that redirects to it with an HTTP 302 status code [40]. For this experiment, I only report the redirection URLs (i.e., the URLs that could serve as lures in a phishing e-mail). I configured each phishing website with the same evasion technique as Experiment B in the respective deployment.

In the first of the three batches, I used a popular link shortening service, *bit.ly*, to generate each of the redirection links. Such services are commonly used by attackers to scalably generate unique lures. In the second of the three batches, I used my own *.com* domains for the redirection links. In the third batch, I also used *.com* domains for the redirection links, but I additionally configured them with server-side IP and hostname cloaking [102]. The latter batch thus most closely mirrors the typical configuration of the phishing websites that I observed in Section 5.5.1. I based the cloaking technique in this batch on the filtering directives found in a *.htaccess* file[4] from a prolific phishing kit that I was able to download from a live phishing website during the PhishTime analysis (❸).

Because I only change one variable between the three batches, I can compare the blacklisting of phishing websites that combine redirection with cloaking on *both* the lure and the phishing website with the blacklisting of websites with lesser degrees of evasion. I can also evaluate the feasibility for attackers to use, and the ecosystem's mitigation of, third-party redirection services.

**Experiment D: Domain Reuse.** After the completion of each Experiment C deployment, I generate websites with different *URL paths* [10], but on the same domains as the

[4]Interestingly, the contents of this *.htaccess* file were highly similar to one I originally analyzed in Chapter 3. Thus, we observe that even advanced phishing kits might re-use old implementations of cloaking techniques.

websites in Experiment C. I then redeploy these websites as part of a new experiment, which seeks to measure how blacklist speed and coverage changes when phishers re-use domains and infrastructure to carry out successive attacks (a strategy intended to increase phishers' ROI).

**Experiment F: Emerging Evasion.** These websites mirror the sophisticated, emerging evasion techniques I observed in Section 5.5.2. Three batches implement evasion using JavaScript code that I found in the wild for CAPTCHA, popup, and mouse movement cloaking, respectively. Three additional batches have the same configuration but with added *.htaccess* server-side cloaking, as in Experiment C. One final batch had only *.htaccess* cloaking, as a control group.

### *5.6.2 Other Measurements*

The remaining experiments (E and G) follow a different reporting methodology than those in the previous section.

**Experiment E: Discovery.** In this experiment, I launch two batches of websites, per deployment, which mirror configuration of Experiments A and B. However, I only report each batch to a single anti-phishing entity (PayPal or the APWG), alternating between deployments. Thus, by comparing against Experiments A and B, I can evaluate how well my primary reporting methodology ensures prompt *discovery* by blacklists. I can also directly test the performance of specific anti-phishing entities: I chose PayPal's own anti-phishing system because my websites used the PayPal brand, and I chose the APWG because it had been shown to reliably share phishing URLs with other entities [7, 101].

**Experiment G: Evidence-Based Reporting.** At the time I initially designed my experiments, Google Safe Browsing only allowed the submission of bare URLs when reporting phishing (whether manually or programmatically). However, in July 2019, with the release of the Chrome Suspicious Site Reporter (CSSR) [23] plugin, manual reports could

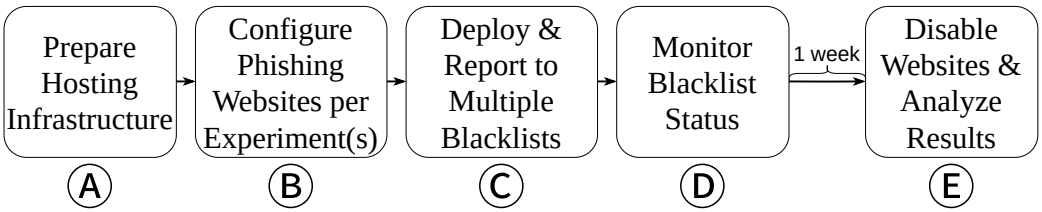



Figure 5.3: Steps in each deployment.

now be enhanced with additional evidence: a screenshot, the redirection chain for the request, as well as the IP address and user agent used to make the request. To evaluate if this enhanced reporting approach could help improve blacklists' detection of evasive URLs, I designed this additional experiment to compare the coverage of GSB when reporting with the old and the new method.

I configured the two batches of phishing websites in this experiment with cloaking that limits traffic to US IP geolocations: a strategy which, as we observed in the previous chapter, was capable of evading GSB [101]. I reported one batch via CSSR [23] and the other batch to GSB via the traditional URL submission form [48]. Because CSSR only supports manual submissions, I compared it to another manual submission channel.

## 5.7 Implementation of Experiments

I leveraged PhishFarm (as outlined in the previous chapter) as the testbed for deploying the phishing websites needed for each of the *PhishTime* experiments [101]. The testbed enables automated configuration, deployment, and monitoring of *innocuous* but real-looking phishing websites to empirically measure the anti-phishing ecosystem.

### *5.7.1 Overview*

In Figure 5.3, I provide an overview of the steps I took to deploy each experiment. First, I prepare the hosting infrastructure (Ⓐ). I used the aforementioned testbed to host

my phishing websites across a total of 42 cloud-based Apache web servers. At the time of each deployment, I configure DNS records to point the required domains to these web servers, and I install Let's Encrypt SSL certificates for each domain. Next, I configure the phishing website content and behavior (i.e., evasion techniques) for each URL, and I test this configuration to verify the correct operation of the framework (Ⓑ). I then activate the websites and immediately report their URLs to the anti-phishing entities specified by the experimental design (Ⓒ). Over the course of the next seven days, I monitor the blacklist status of my URLs and I collect web traffic metadata (Ⓓ). Finally, I deactivate the websites and analyze the data that has been collected (Ⓔ). Although I supervised each deployment, each of these steps is otherwise fully automated.

My phishing websites matched the look and feel of the *PayPal.com* login page as it appeared in January 2019. Whenever a crawler visit was subject to the cloaking technique in use by a particular website, I would display a generic 404 error message [40], as shown in Figure 5.4. To avoid confounding effects, I did not change the appearance of my websites between deployments, and I used randomized, non-deceptive URLs for each. Possible implications of this are discussed in Section 5.12.

### *5.7.2 Reporting to Blacklists*

To allow for automated experiments and to maintain consistency across deployments, I fully automated my reporting methodology. My approach is representative of the actions that an organization targeted by phishers might take to mitigate known phishing websites [101].

To report each of my phishing websites, I submit its URL directly to Google Safe Browsing via the Phishing Protection Submission API [46][5] and to the APWG via the

[5]At the time of my deployments, the Phishing Protection Submission API was in a beta stage and not available to the public. Google provided us with access to the API for this research.

eCrime Exchange API [7]. Direct API reporting is not available for Opera and Microsoft SmartScreen. However, prior work has shown that the APWG and other major anti-phishing entities share data with these blacklists [101, 111]. Therefore, I report to these additional entities via e-mail. Using a PayPal-branded phishing e-mail template found in the wild, I generate a fake phishing e-mail with the URL of the website. I then forward this e-mail as an attachment to top anti-phishing entities that accept reports from the public: PhishTank [107], Netcraft [97], PayPal [105], and US CERT [43]. This reporting methodology seeks to guarantee all blacklists' discovery of my phishing websites (thus, it does not apply to Experiments E and G, as previously discussed in Section 5.6.2).

### *5.7.3 Blacklist Monitoring*

For my blacklist metrics, I used a total of 40 virtual machines (VMs) to empirically monitor blacklisting of each website at least once every ten minutes across six desktop browsers: Chrome, Safari, Firefox, Edge, Internet Explorer, and Opera. To determine the speed of blacklisting on mobile, I monitor Google Safe Browsing programmatically using the Update API [48]. Using a single physical Android device (connected to the Internet over Wi-Fi), I also empirically compared the coverage of mobile Chrome, Firefox, and Opera to their desktop counterparts.

I used one VM to check if any of my URLs had been taken down (mainly to monitor *bitly.com* links), and another to check my SSL certificates against Let's Encrypt's certificate revocation list.

Beyond measuring the time of initial blacklisting of each URL, I also wanted to measure the persistence of blacklisting and possible de-blacklisting. The system, therefore, continued monitoring every URL for the full duration of each experiment, and for a one-week period afterward.

### 5.7.4 *Infrastructure Changes*

To support the expanded requirements of PhishTime, I made several additions to the PhishFarm testbed.

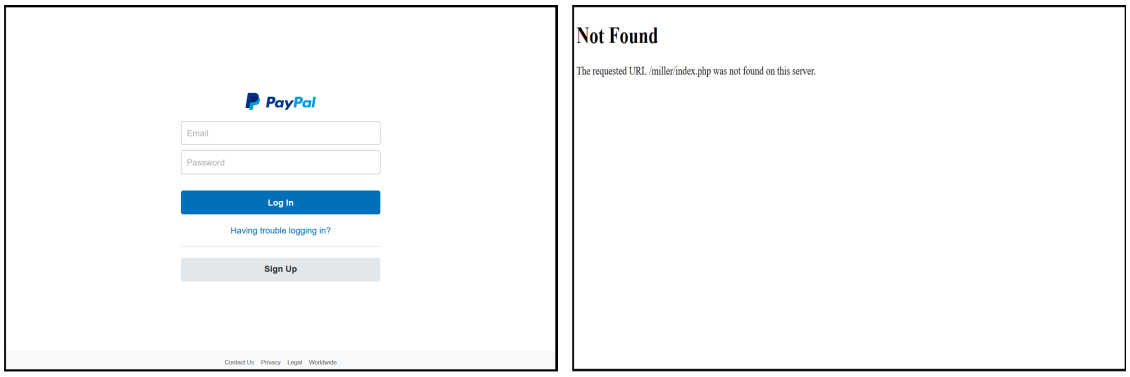
**SSL Certificates:** As of 2019, the majority of phishing websites in the wild have switched to using HTTPS [7]. Therefore, I mirror this trend in my research and developed a series of scripts to automatically provision Let's Encrypt certificates for my phishing websites. Although Let's Encrypt does not claim to revoke certificates tied to malicious websites [69], to verify, I additionally developed functionality to check my certificates' revocation status via OCSP [93].

**Redirection Links:** Because many of my experiments involve the use of redirection links, I added a framework module for automatically configuring and monitoring redirection URLs that point to specific batches of my own phishing URLs. In addition to using dedicated domains for redirection, third-party redirection services are also supported. In the latter case, the monitoring system checks to see if the service has taken down the URL (i.e., by marking it as malicious and disabling the normal redirection).

**Google Safe Browsing API:** To eliminate the need for manual reporting in my longitudinal measurements, I added a module for programmatically reporting URLs to Google via the GSB API. To facilitate the scalable monitoring of mobile blacklisting, as discussed in the previous section, I also added functionality to programmatically (and anonymously) monitor the GSB blacklist status of my websites.

**PhishTime Analysis:** To adapt the PhishFarm framework to monitor phishing websites in the wild (as required by PhishTime), I made several changes to the Research Client, as originally introduced in Section 4.4.2 in Chapter 4. With these changes, the client gained an *ecosystem monitoring mode* which facilitates the reporting of unmitigated phishing websites to blacklists and performs subsequent monitoring of blacklist status.

### 5.7.5 *Configuration and Experimental Controls*




(a) Successful request.

(b) Request denied by cloaking.

Figure 5.4: Appearance of the phishing websites used in the PhishTime experiments.

I took care to ensure the reliability of my experimental data by precisely controlling numerous aspects of my website configuration and deployment.

**Eliminating Confounding Effects.** To the extent possible, I sought to prevent any external factors—such as the high volume of monitoring traffic or content of my websites' URLs—from skewing my blacklist measurements. Therefore, I configured my websites to never render phishing content for requests originating from my monitoring infrastructure's IP addresses.

To ensure that no prior malicious activity would impact my phishing URLs, I registered a unique (and thus previously unreported) domain name for each URL (with the exception of Experiment D, in which I deliberately measured domain re-use). My domains and URL paths consisted of combinations of random English words to limit the effect of heuristics and URL fingerprinting [138] in successive deployments of my websites. I discuss certain trade-offs of this decision in Section 5.12.

**Experimental Variables.** Within each experiment, I varied the configuration of different batches in at most one way, such that I could focus the comparative analysis

on a single variable. The same concept also applies *between* the majority of my experiments, which can thus collectively paint a multi-dimensional view of the response of anti-phishing blacklists.

**Reporting.** To avoid bias, each of my e-mail reports originated from a different e-mail address, and information such as the (fictitious) victim name or transaction amount was randomized within each e-mail. I also throttled the reports to avoid an excessive reporting rate.

**Experiment Duration.** Anti-phishing blacklists typically respond within a matter of hours [103]. However, in certain cases (e.g., due to cloaking), blacklisting may be delayed by several days [101]. This observation, combined with occasional long-lasting phishing websites during the PhishTime analysis, motivated my conservative choice of a one-week lifespan of each phishing website within my experiments.

**Testing.** Before starting each deployment, I tested the availability of my domains, servers, monitoring infrastructure, and reporting infrastructure to prevent any unanticipated technical challenges.

**Preliminary Deployment.** Before the first deployment of the main experiments, I carried out a preliminary deployment with batches of websites configured as defined for Experiments A through F. This deployment used its own set of 864 URLs with 486 unique domain names.

The purpose of this deployment was to verify the soundness of my experimental design and the technical correctness of my testbed, such that I could make any necessary changes to either prior to the subsequent deployments. Thus, I maximize the quality of my experimental data while ensuring that the many domains purchased for this research are not wasted.

The preliminary deployment showed that phishing websites configured with the evasion techniques I had chosen generally reduced blacklist speed compared to the baseline,

and that my reporting methodology resulted in the prompt discovery of my URLs by all the blacklists being tested. Therefore, I did not make changes to the experimental design. However, this deployment also prompted me to make several optimizations to the testbed software to improve the accuracy of blacklist monitoring in the long term. I thus focus my evaluation only on the main deployments.

## 5.8 Experimental Results

| | | **Desktop** | | | | | | **Mobile** | | | | **Avg. Traffic** | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | *GSB* | | *SmartScreen* | | *Opera* | | *GSB: Chrome/Safari* | | *GSB: Firefox* | *Opera* | *All* | 200 |
| **Deployment** | | Coverage | Median Speed | Coverage | Median Speed | Coverage | Median Speed | Coverage | Median Speed | Coverage | Coverage | | |
| *1* | May 2019 | 100.0% | 00:44 (hh:mm) | 100.0% | 02:02 | 98.1% | 00:37 | 100.0% | 09:19 | 100.0% | 0.0% | 1677 | 1151 |
| *2* | Jul. 2019 | 100.0% | 00:51 | 100.0% | 02:38 | 70.4% | 00:32 | 55.6% | 35:28 | 100.0% | 0.0% | 7003 | 1491 |
| *3* | Sep. 2019 | 64.8% | 00:50 | 61.1% | 04:44 | 22.2% | 01:52 | 13.0% | 159:22 | 64.8% | 14.8% | 286 | 211 |
| *4* | Oct. 2019 | 98.1% | 01:00 | 100.0% | 02:19 | 64.8% | 00:55 | 50.0% | 03:05 | 98.1% | 14.8% | 3756 | 2020 |
| *5* | Nov. 2019 | 100.0% | 01:26 | 100.0% | 02:27 | 59.3% | 00:38 | 13.0% | 39:11 | 100.0% | 0.0% | 1566 | 682 |
| *6* | Dec. 2019 | 100.0% | 00:46 | 100.0% | 02:34 | 48.1% | 00:28 | 70.4% | 00:28 | 100.0% | 9.3% | 3255 | 1554 |

Table 5.2: Blacklist performance vs. unevasive phishing (Experiment A: raw data for each deployment).

| | | **Desktop** | | | | | | **Mobile** | | | | **Avg. Traffic** | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | *GSB* | | *SmartScreen* | | *Opera* | | *GSB: Chrome/Safari* | | *GSB: Firefox* | *Opera* | *All* | 200 |
| **Deployment** | | Coverage | Median Speed | Coverage | Median Speed | Coverage | Median Speed | Coverage | Median Speed | Coverage | Coverage | | |
| *1* | May 2019 | 89.5% | 02:29 (hh:mm) | 88.0% | 10:20 | 83.4% | 03:54 | 54.9% | 07:36 | 89.5% | 0.0% | 2038 | 603 |
| *2* | Jul. 2019 | 99.3% | 01:46 | 99.5% | 05:42 | 43.0% | 01:41 | 0% | - | 99.3% | 31.8% | 508 | 53 |
| *3* | Sep. 2019 | 79.9% | 02:21 | 69.5% | 08:24 | 50.1% | 02:36 | 3.2% | 34:47 | 79.9% | 29.3% | 1073 | 589 |
| *4* | Oct. 2019 | 86.7% | 01:32 | 90.0% | 10:19 | 58.2% | 01:51 | 0% | - | 86.7% | 35.0% | 545 | 45 |

Table 5.3: Blacklist performance vs. evasive phishing (Experiments B, C, D: average of all deployments).

After the completion of all my experiment deployments, I had collected extensive data for each of the 4,158 URLs that I launched and monitored: timestamps of blacklisting (in six desktop browsers, three mobile browsers, and the Google Safe Browsing API), online status, certificate revocation status, and web traffic logs. The infrastructure operated as

expected during each main deployment and, thus, all the corresponding data was valid for analysis.

In the analysis that follows, for any batch of URLs, I define the *coverage* of a given blacklist as the percentage of all URLs that were blacklisted *at any point* during the seven-day deployment of the batch. For any given URL, I define blacklist *speed* as the time elapsed between *my* reporting of that URL and its subsequent blacklisting. Within an individual batch, I either provide the median speed in tabular form or plot speed as a function of coverage over time.

**Simplification of Dimensionality:** My empirical blacklist monitoring of desktop browsers revealed that Chrome and Safari consistently delivered the same speed and coverage, whereas Firefox was an average of ten minutes slower (stemming from different caching of the GSB Update API [46]) but still had the same coverage. Similarly, in comparing Edge and Internet Explorer across all of my deployments, I found that the latter was 12 minutes slower on average, also with the same coverage. Thus, to simplify and clarify my analysis, I exclude the desktop versions of Safari, Firefox, and Internet Explorer from my evaluation.

On mobile devices, I found that Firefox had identical blacklist coverage as its desktop counterpart (an improvement which resulted from the PhishFarm disclosures in Chapter 4). However, neither mobile Chrome nor mobile Opera proved to be consistent with their desktop versions. Therefore, I evaluate data from the latter two browsers alongside mobile Firefox.

**Data Aggregation:** I aggregate my blacklist measurements based on the objectives of each experiment, as defined in Section 5.6. For longitudinal comparisons, I group blacklist performance by deployment; to evaluate evasion, I aggregate multiple deployments by experiment or batch.

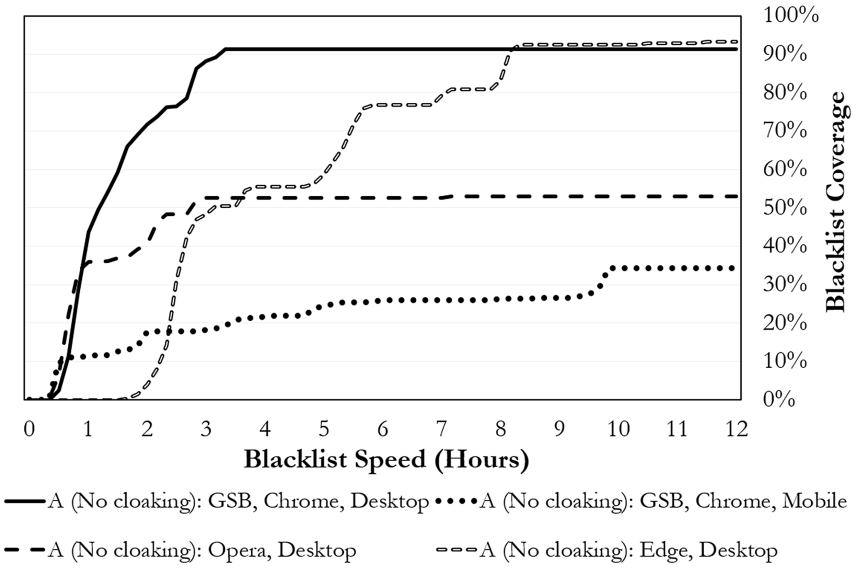



Figure 5.5: Aggregate speed and coverage of all blacklists across Experiment A (Deployments 1-6).

### *5.8.1 Discovery*

Of the 4,158 URLs that I deployed, 4,068 received traffic from at least one crawler. The 94 URLs which were never visited were all part of Deployment 3: 81 URLs were part of Experiment E (reported to a single entity) and 13 were landing pages within Experiment C.

Of all my URLs, a total of 3,514 were blacklisted in at least one browser. Of the 644 URLs which were never blacklisted, 299 were part of Experiment F (in which sophisticated cloaking successfully evaded detection), 131 were part of Experiments E or G (which were not designed to guarantee discovery), and 214 were part of Experiments B, C, and D (with cloaking and redirection).

Given that the aforementioned lack of traffic can be attributed to the ecosystem issues I identified during Deployment 3 (discussed in Section 5.8.2), and the fact that all websites from Experiment A were blacklisted in at least one browser, I believe that my reporting methodology was successful in ensuring prompt discovery by the ecosystem.

| | | Desktop | | | | | | Mobile | | | | Avg. Traffic | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | *GSB* | | *SmartScreen* | | *Opera* | | *GSB: Chrome/Safari* | | *GSB: Firefox* | *Opera* | *All* | 200 |
| **Experiment** | **Batch** | Coverage | Median Speed | Coverage | Median Speed | Coverage | Median Speed | Coverage | Median Speed | Coverage | Coverage | | |
| *Experiment A (Baseline)* | | 92.9% | 00:57 (hh:mm) | 93.2% | 02:48 | 60.5% | 00:55 | 57.8% | 17:30 | 92.9% | 3.7% | 3452 | 1366 |
| *Experiment B (Basic Evasion)* | JavaScript Cloaking | 88.3% | 01:03 | 100.0% | 03:30 | 49.4% | 00:56 | 0.0% | - | 88.3% | 0.0% | 455 | 115 |
| | Mobile Cloaking | 100.0% | 00:55 | 100.0% | 02:39 | 44.0% | 00:38 | 0.0% | - | 100.0% | 0.0% | 936 | 207 |
| Experiment C (Typical Evasion - Redirection) | bit.ly Redirection - Lure | 86.1% | 01:25 | 91.4% | 03:02 | 46.3% | 01:40 | 0.0% | - | 86.1% | 0.0% | 2313 | 2313 |
| | bit.ly Redirection - Landing | 86.1% | 02:58 | 88.0% | 12:45 | 59.3% | 02:46 | 25.9% | 43:51 | 86.1% | 25.0% | 593 | 392 |
| | .com Redirection - Lure | 83.3% | 01:44 | 99.4% | 03:09 | 50.6% | 01:57 | 0.0% | - | 83.3% | 41.4% | 440 | 81 |
| | .com Redirection - Landing | 88.9% | 02:48 | 87.0% | 09:35 | 59.7% | 02:55 | 24.1% | 11:46 | 88.9% | 30.6% | 740 | 454 |
| | .com Redirection w/ .htaccess | 80.2% | 01:36 | 77.2% | 08:51 | 37.7% | 01:31 | 0.0% | - | 80.2% | 25.3% | 275 | 28 |
| | .com Redirection w/ .htaccess - Landing | 84.3% | 02:43 | 86.6% | 10:01 | 51.9% | 02:33 | 9.3% | 11:19 | 84.3% | 32.9% | 370 | 63 |
| Experiment D (Domain re-use) | bit.ly Redirection - Lure | 96.3% | 01:09 | 94.4% | 06:51 | 58.0% | 00:41 | 0.0% | - | 96.3% | 0.0% | 5143 | 5143 |
| | bit.ly Redirection - Landing | 97.2% | 02:03 | 72.7% | 11:56 | 58.0% | 02:21 | 4.3% | 00:01 | 97.2% | 52.5% | 876 | 497 |
| | .com Redirection - Lure | 95.7% | 01:10 | 99.4% | 06:59 | 73.5% | 27:24 | 0.0% | - | 95.7% | 54.9% | 1582 | 984 |
| | .com Redirection - Landing | 98.1% | 02:10 | 71.3% | 11:48 | 66.7% | 01:50 | 3.7% | 46:28 | 98.1% | 50.0% | 1061 | 534 |
| | .com Redirection w/ .htaccess - Lure | 93.8% | 01:13 | 77.2% | 10:07 | 37.7% | 00:57 | 0.0% | - | 93.8% | 37.0% | 1051 | 583 |
| | .com Redirection w/ .htaccess - Landing | 95.4% | 02:17 | 67.3% | 12:19 | 45.7% | 01:53 | 0.0% | - | 95.4% | 40.7% | 332 | 42 |
| Experiment E (Discovery) | Reported to APWG | 98.1% | 02:47 | 100.0% | 02:29 | 41.7% | 02:41 | 51.9% | 04:53 | 98.1% | 41.7% | 2901 | 1591 |
| | Reported to PayPal | 16.2% | 01:06 | 38.4% | 02:43 | 6.5% | 00:49 | 13.0% | 00:35 | 16.2% | 2.8% | 450 | 293 |
| Experiment F (Emerging Evasion) | Mouse Movement Cloaking | 0.0% | - | 0.0% | - | 0.0% | - | 0.0% | - | 0.0% | - | 37 | 34 |
| | CAPTCHA Cloaking | 0.0% | - | 42.6% | 03:06 | 0.0% | - | 0.0% | - | 0.0% | - | 47 | 42 |
| | Notification Cloaking | 0.0% | - | 0.0% | - | 0.0% | - | 0.0% | - | 0.0% | - | 48 | 41 |
| | .htaccess Cloaking | 100.0% | 01:37 | 100.0% | 10:47 | 42.6% | 00:40 | 0.0% | - | 100.0% | 0.0% | 702 | 86 |
| | .htaccess Cloaking & Mouse Movement Cloaking | 0 coverage | | | | | | | | | | 59 | 20 |
| | .htaccess Cloaking & CAPTCHA Cloaking | | | | | | | | | | | 45 | 19 |
| | .htaccess Cloaking & Notification Cloaking | | | | | | | | | | | 48 | 21 |
| Experiment G (Reporting Methods) | Standard URL Report | 20.4% | 00:38 | 0.0% | - | 0.0% | - | 20.4% | 00:17 | 20.4% | 0.0% | 5 | 2 |
| | Chrome Suspicious Site Reporter | 90.7% | 10:13 | 0.0% | - | 0.0% | - | 90.7% | 10:17 | 90.7% | 0.0% | 16 | 14 |

Table 5.4: Blacklist performance aggregated by each batch (average of all deployments).

### *5.8.2 Overall Blacklist Performance*

In Table 5.2, I show the blacklist speed and coverage results from each of the six deployments of Experiment A, as well as the average number of crawler requests to *each* individual website. Because this experiment consisted solely of unsophisticated phishing websites without any form of evasion, it allows us to establish a baseline for best-case blacklist performance which I can compare against other experiments.

**Desktop Blacklists.** With an average coverage of 92.9% and an average (median) speed of 57 minutes across the six deployments, overall, GSB proved to be the best-performing blacklist that I tested. SmartScreen showed a slightly higher coverage of 93.2%,

but had a slower speed of 3 hours and 47 minutes. Opera's coverage was the lowest, at 60.5%, though its 55-minute speed managed to inch ahead of GSB.

**Mobile Blacklists.** The mobile version of Firefox mirrored the 92.9% coverage of GSB on desktop and had the highest coverage of the mobile blacklists I tested. Mobile Chrome and mobile Safari delivered a much lower coverage of 57.8%, whereas Opera's coverage was minimal at 3.7%.

Although aggregate speed and coverage metrics provide an assessment of overall blacklist performance, they fail to illustrate specific differences in behavior between blacklists. In Figure 5.5, I plot the growth of each blacklist's coverage over the course of the first 12 hours of my deployments (the same data over the full one-week deployment is in Figure 5.6). I observe that GSB and Opera both start blacklisting as early as 20 minutes after receiving my phishing reports. On desktop platforms, coverage grows quickly and stabilizes after approximately three hours; on mobile devices, coverage grows more slowly over a longer period. SmartScreen's earliest response occurred about one hour after GSB and Opera, and grew over a seven-hour period thereafter.

**Long-term Blacklist Consistency.** High blacklist speed and coverage are necessary to effectively protect users from phishing websites, but, given the pressure upon the ecosystem by phishers [7, 47], it is equally important that blacklist performance remains consistent in the long term. By comparing the measurements between successive deployments (in Table 5.2 and 5.3 for non-evasive and evasive phishing websites, respectively), I can evaluate this consistency.

Per the data for Experiment A, I observe that both GSB and SmartScreen delivered 100% coverage and similar speed in five of the six deployments. Opera remained consistent in terms of speed across five deployments. Except for GSB in mobile Firefox, blacklists in mobile browsers did not show such consistency, however.

Notably, there was a significant drop in coverage during Deployment 3, both for non-evasive and evasive phishing, as shown in Tables 5.2 and 5.3. In analyzing this anomaly, I first ruled out technical issues in my reporting methodology and confirmed that all of my e-mail and API reports were successfully delivered. I also redeployed Experiment A with prior domains and were able to reproduce degraded coverage. After analyzing the results of Experiment E (summarized in Table 5.4), I found that the coverage from reports sent directly to PayPal was similarly low: its coverage in GSB was 9.3% in Deployment 3 compared to 44.4% in Deployment 1. Upon comparing crawler traffic between these deployments, I found that crawler activity as a result of PayPal reports was absent from the majority of websites in Deployment 3. Although I cannot rule out other ecosystem factors, I believe that this absence was a key cause of the overall coverage drop, and I disclosed this to PayPal (I later received acknowledgment of the issue).

**Blacklist Persistence.** Across all of my deployments, once a URL was blacklisted in a particular blacklist, I did not observe de-blacklisting within the one-week deployment period, or within a one-week period immediately after each deployment. After the conclusion of my final deployment, I monitored the URLs for an extended period of time and found that the earliest removal from blacklists occurred 29 days after I had originally reported the respective URL. During this period, my infrastructure remained online and served 404 errors for all domains from Deployment 6.

I suspect that de-blacklisting may depend on factors such as the presence of benign content on a domain, the domain's reputation, or whether the web server is online. Although my experiments were not designed to pinpoint criteria for de-blacklisting, I believe that premature removal of phishing URLs from blacklists is not a significant issue.

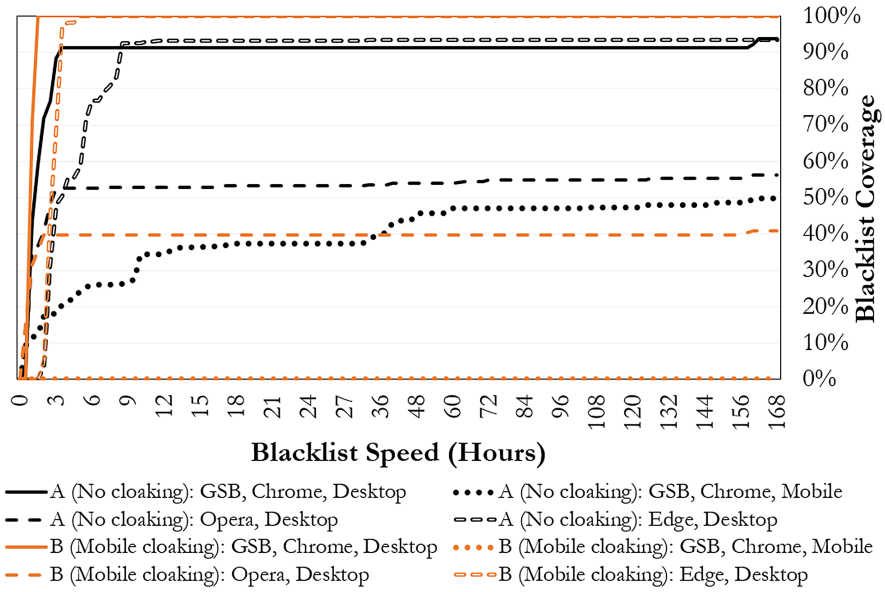



Figure 5.6: Comparison of all blacklists' aggregate performance for uncloaked websites vs. websites with Mobile cloaking.

### *5.8.3 Typical Evasion Techniques*

In Table 5.4, I show blacklist performance for the specific batches within each experiment, aggregated across all deployments. This allows us to compare speed and coverage when blacklists are faced with different evasion techniques.

Websites with only mobile user-agent cloaking (in Experiment B) had a negligible effect on desktop blacklisting compared to Experiment A (if I disregard the skew from Deployment 3): modern blacklists can, in fact, reliably detect phishing websites with certain basic evasion techniques. Interestingly, however, both GSB and Opera had 0% coverage on mobile devices across all deployments of Experiment B, which is very undesirable given that Experiment B websites were configured to be accessible exclusively on mobile devices. In Figure 5.6, I visualize blacklisting of these websites in each blacklist over the full duration of my deployments.

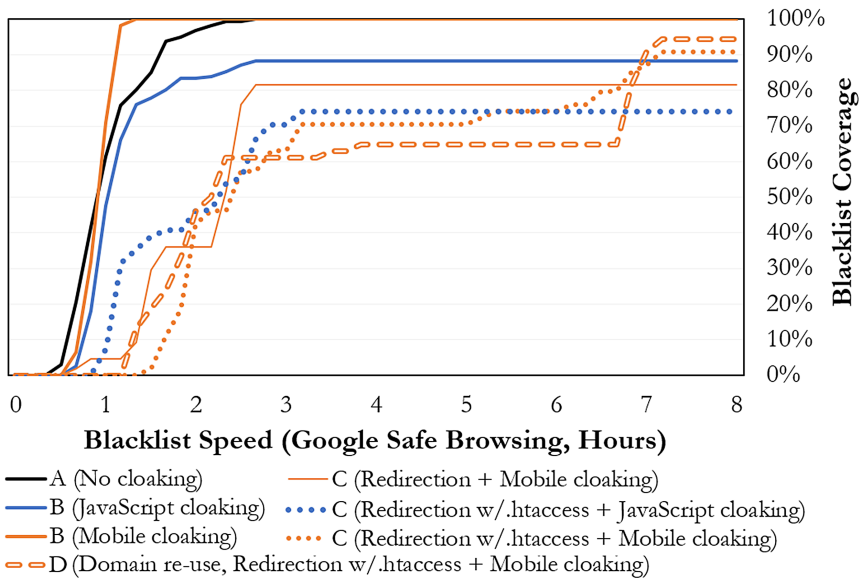



Figure 5.7: Comparison of aggregate speed and coverage of GSB against different evasion techniques.

In Experiment C, I tested the addition of three types of redirection on top of the evasion techniques in Experiment B. For brevity, I focus my discussion on blacklisting by GSB on desktop, and I use the average speed and coverage across all deployments of Experiment B (00:59 and 94.2%, respectively), calculated per Table 5.4, in the following comparisons. On average, redirection via *bit.ly* links slowed blacklisting speed of the corresponding landing pages to 02:58, and reduced coverage to 86.1%. Redirection via *.com* domain names slowed the speed to 02:48 and reduced coverage to 88.9%. By adding *.htaccess* cloaking on top of redirection, the speed only slowed to 02:43, but coverage fell further to 84.3%. As shown in Table 5.4, the speed of blacklisting of the corresponding lures was at least one hour faster in each case; however, attackers' ability to easily generate many lures places an increased importance on the blacklisting of actual landing pages [130].

In Experiment D, I re-deployed phishing websites on the same *.com* domains as in Experiment C, but with different paths, to simulate how attackers might re-use compromised domains in the wild. Although I observed increased speed and coverage compared to Ex-

periment C, the speed remained slower than in experiments without redirection. Only 4.3% of URLs in Experiment D were blacklisted immediately upon deployment, which may represent a gap exploitable by phishers. In Figure 5.7, I visualize the difference in GSB desktop blacklisting of the cloaking techniques considered in this section. To maintain consistency, I exclude Deployment 3 from the figure. For clarity, I also omit the batches with *bit.ly* links, as they followed the same trend as *.com* redirection links, and were only blacklisted slightly more slowly.

In mobile Chrome and Safari, Experiment C coverage ranged from just 9.3% to 25.9% and was 8 to 40 hours slower than on desktop. Only landing pages, rather than lures, were blacklisted. Interestingly, in Experiment D, coverage dropped to a range of 3.7% to 4.3%, despite the ecosystem's knowledge of my domains from previously blacklisted URLs. I discuss the implications of these findings in Section 5.10.

Overall, I observe that delays and gaps exist in the blacklisting of typical phishing websites: these gaps provide a prime opportunity for attackers to successfully target their victims [103], help explain the prevalence of evasion techniques and should be addressed by the ecosystem.

**Disabling of Bit.ly Links.** To deter abuse, *bit.ly* disables redirection links which point to known malicious content. During Deployment 1, *bit.ly* disabled 98.1% of the links within Experiment C, and 88.9% of the links within Experiment D, with an average speed of 11 hours and 36 minutes (far slower than the tested blacklists). However, except for a single URL during Deployment 3, no other URLs were disabled over the course of this research. I disclosed these findings to *bit.ly* but did not receive a response.

### *5.8.4 Emerging Evasion Techniques*

As shown in Table 5.4, none of the batches of sophisticated cloaking techniques within Experiment F saw any blacklisting, with the exception of one batch with CAPTCHA cloak-

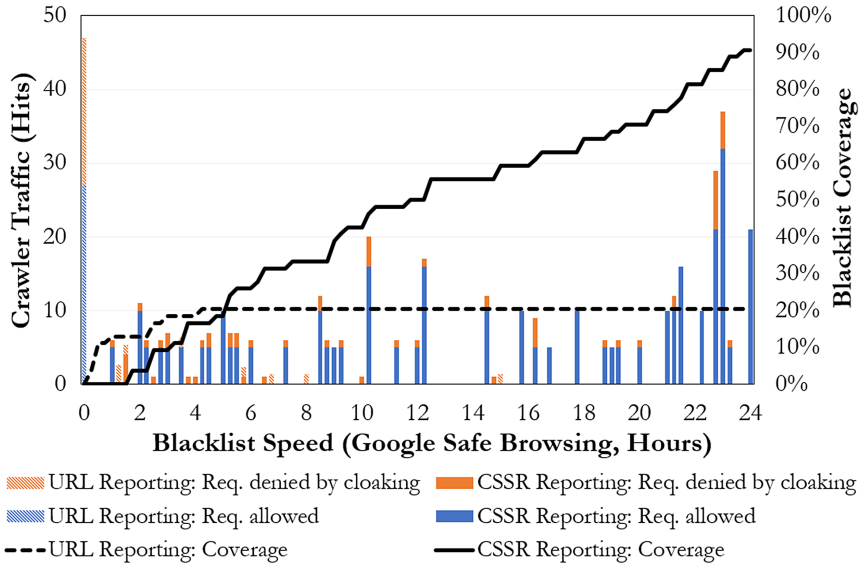



Figure 5.8: Comparison of traditional URL-only reporting with *evidence-based* reporting in Google Chrome.

ing, which had 42.6% coverage in SmartScreen only. Upon further investigation, I discovered that SmartScreen's detection occurred due to its classification of obfuscation within the CAPTCHA JavaScript code as malware. Because such detection can trivially be bypassed [139], I believe that behavior-based evasion techniques represent a threat to the anti-phishing ecosystem.

A fundamental distinction between the cloaking techniques in this experiment and the other experiments is that they require *interaction* from a human user to trigger the display of phishing content (i.e., clicking on a button, solving a CAPTCHA challenge, or moving the mouse). Such behaviors might be typical of a human user (and may not even raise suspicion if the user is successfully fooled by an e-mail lure, or if the landing page matches the look-and-feel of the impersonated organization). However, web crawlers would need to be specially developed to emulate such behaviors or otherwise fingerprint such cloaking.

### 5.8.5 *Single-entity Reporting*

In Experiment E, I observed clear differences in blacklist response when comparing reporting to the APWG (only) with reporting to PayPal (only), as shown in Table 5.4. Even if I exclude the problematic performance of PayPal during Deployment 3 (as discussed in Section 5.8.2), reporting to the APWG resulted in higher coverage across all blacklists and greater crawler traffic to each website. However, the speed of GSB blacklisting after reporting to PayPal was 01:31 faster than that of the APWG. This suggests that between entities, there exist different implementations for propagating reported URLs to blacklists. Due to each entity's unique strengths, I believe it is important to report phishing to multiple entities.

### 5.8.6 *Evidence-based Reporting*

In Figure 5.8 and Table 5.4, I compare the difference in GSB speed and coverage between traditional URL reporting and evidence-based reporting through CSSR [23] (note that I limit the *x-axis* as coverage did not increase after 24 hours), as measured in Experiment G.

I observe that the two reporting approaches each resulted in a distinct crawler response and subsequent blacklist performance. Traditional URL reporting was followed by an immediate burst of crawler traffic and a negligible amount of crawler traffic in the hours thereafter. Even though 50% of the phishing websites I reported were successfully retrieved by a crawler, only 20.4% were ultimately blacklisted. The earliest blacklisting occurred 20 minutes after reporting, and coverage stopped growing after approximately 4 hours. Reporting through CSSR yielded a slower initial speed, but resulted not only in 90.7% coverage within 24 hours, but also a substantially higher amount of crawler traffic, spread over a long period of time, with 47.5% fewer requests being denied by cloaking.

The earliest blacklisting occurred 50 minutes after reporting, and coverage matched that of the traditional reporting approach by the 4-hour mark.

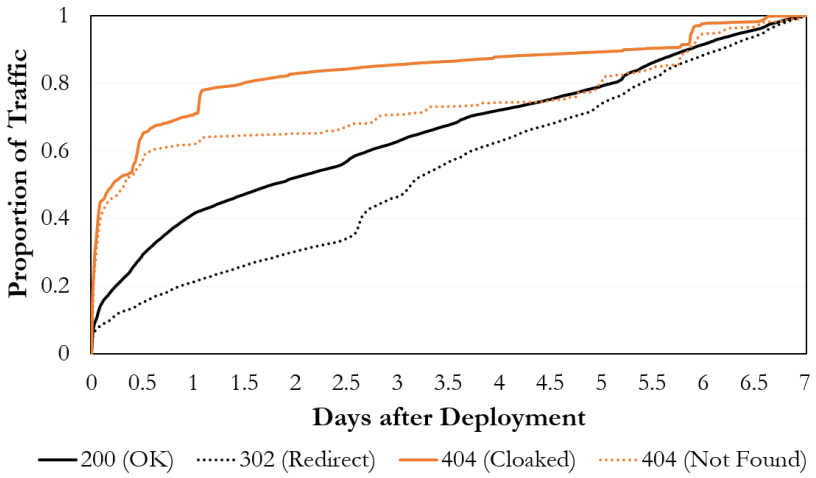



Figure 5.9: Cumulative Distribution Function (CDF) plot of traffic to my phishing websites.

## 5.9 Crawler Traffic

Between May 2019 and January 2020, the 4,158 URLs that were part of my main deployments received a total of 2.14 million HTTP requests from 41,750 distinct web crawler IP addresses. An additional 20.50 million requests were made from my monitoring infrastructure to check the blacklist status of my websites. My websites remained online for the duration of the deployments (i.e., were not subjected to take-down [1]) as I had made my hosting provider aware of the nature of my research.

55.27% of the crawler requests were successful and returned an *HTTP 200* status code (*302* for redirection links). The remaining requests returned an *HTTP 404* status code: 7.56% were denied by a cloaking technique, and 37.18% simply tried to fetch nonexistent URLs. The majority of the nonexistent URLs represented crawler efforts to scan for phishing kit archives or credential dump files, which is a common way not only to fingerprint

phishing websites, but also to identify stolen credentials which may linger on the same server as the phishing kit [22].

In Figure 5.9, I show the Cumulative Distribution Function (CDF) of crawler traffic to my websites. I observe that after an initial burst within the first day of deployment, successful traffic remains fairly consistent for the remainder of the deployment. This traffic accounts for crawlers which continuously monitor the presence of the phishing website.

The relative proportion of requests which are denied through cloaking drops over time. The increased effort early on allows crawlers to fingerprint evasion techniques such that future requests are more likely to be successful. I believe that this behavior in part helped blacklists deliver the high coverage that I observed, even for websites with more sophisticated cloaking techniques as in Experiment C.

## 5.10 Discussion

Although blacklists are capable of *detecting* the typical evasion techniques which I tested—including cloaked redirection—the PhishTime experiments showed that these techniques generally both slow speed and reduce coverage. Moreover, in the wild, attackers use redirection chains far more complex than those I tested. Therefore, in practice, coverage of phishing websites hidden behind redirection may be worse than what was observed in my experiments. Other notable gaps in coverage also remain, particularly on mobile devices. Given attackers' ability to adapt to the ecosystem by leveraging innovative evasion strategies, such as those in Experiment F, I believe that evasion remains a key anti-phishing concern.

**Defensive Strategy.** To the best of my knowledge, longitudinal measurements of anti-phishing defenses are not currently being performed at the ecosystem level. The PhishTime framework, combined with deployments of targeted experiments, can be used as a defensive strategy to identify gaps in defenses and help address them through security

recommendations. Although my work focuses on browser blacklists, the scope of future experiments could be shifted to evaluate other mitigations (e.g., spam filters). Moreover, the ecosystem analysis could be aimed at areas other than evasion techniques, such as identifying attacker-friendly web hosts or compromised domains [1].

Depending on the objective of the entity carrying out the experiments, PhishTime can be used to assess aspects of the ecosystem as a whole, or the behavior of a specific entity or mitigation. I believe this is a crucial first step toward achieving consistency in—and perhaps accountability for—anti-phishing and account protection [125] efforts of the many different organizations that phishers impersonate. I have proposed this approach to the APWG; subsequently, efforts are underway to incorporate PhishTime as an ecosystem-level service which can be used to monitor URLs reported to the APWG eCrime exchange and drive future experiments based on this dataset.

**Reporting Protocols.** Given the prevalence of evasive phishing in the wild, and the promising performance of CSSR, I believe that the adoption and expansion of evidence-based reporting protocols should be a priority for the ecosystem. In addition to supporting manual reporting by humans, such protocols should be made available to vetted automated anti-phishing systems. A key benefit of such an integration would be that if one entity *detects* an evasive phishing website, it can share the parameters used for detection to help other entities (e.g., blacklists) avoid duplication of effort while improving mitigation (e.g., speed and coverage). Moreover, such evidence can be used to support take-down efforts [1] or law enforcement intervention if an initial mitigation, such as blacklisting, proves insufficient.

Beyond the expansion of enhanced reporting protocols, I believe that standardized methods for reporting phishing *across* the ecosystem—rather than to individual entities—would help improve the ecosystem's collective response. As I observed from Experiment E, each anti-phishing entity functions differently and, thus, affects blacklisting differently.

Additionally, the drop in coverage I observed during Deployment 3 suggests that the ecosystem may in some cases be fragile. If a single anti-phishing entity contributes disproportionately to the mitigation of a particular type of threat, it can become a *chokepoint*, which, in case of a temporary failure or degradation, could represent an opportunity for phishers to launch a burst of successful attacks.

**Mobile Blacklists.** Mobile users account for a majority of Internet traffic [120], and prior research has shown that mobile web browsers are particularly prone to phishing attacks [73]. Yet, my findings indicate that the anti-phishing protection in mobile web browsers continues to trail behind that of desktop browsers. The bandwidth used by mobile devices—which may be subject to mobile carrier restrictions—was historically a barrier to desktop-level GSB protection in mobile Chrome and Safari [101]. However, over a Wi-Fi connection (which I used for monitoring), the full blacklist should be checked.

My experiments were unable to determine exactly how blacklists determine if a URL blacklisted in desktop browsers should also be blacklisted in the corresponding mobile browser. However, I did observe that my websites which were deliberately configured to only be accessible in mobile browsers (i.e., Experiment B) were in fact never blacklisted in these browsers. I, therefore, believe that mobile blacklisting represents a key vulnerability within the ecosystem and that this blacklisting should be made consistent to better protect mobile users targeted by phishers.

**Certificate Revocation.** Throughout my deployments, I monitored the OSCP revocation status [93] of my domains' SSL certificates, which I automatically obtained from Let's Encrypt (a free Certificate Authority with the highest representation among phishing websites in the wild [31]). *None* of the certificates were revoked. In addition, I found that certificates could also be issued for domains that were already blacklisted, as Let's Encrypt had discontinued checking of domains in new certificate requests against GSB in early 2019 [69]. Although the role of Certificate Authorities as a mitigation against

phishing is subject to debate [31], I believe that the ease at which attackers can obtain certificates represents a noteworthy security consideration.

**Ecosystem Changes.** A notable ecosystem development took place in December 2019: in Chrome 79, Google improved the speed of GSB by "up to 30 minutes" [45] by incorporating real-time lookups of phishing URLs for users of Chrome's password manager. Although not yet enabled by default, this change acknowledges, and seeks to address, the delay to blacklisting speed possible in the default implementation of GSB, which caches and periodically updates a local copy of the URL blacklist. Due to the timing of the release of this feature, I was not able to evaluate it in my experiments.

## 5.11 Ethical Concerns

I sought to address a number of potential ethical concerns while conducting this research.

**Disclosures.** Beyond the disclosures discussed throughout Section 5.8, upon the completion of my final deployment, I sent a report with my findings to PayPal, Google, Microsoft, Apple, and Opera, with a particular focus on the emerging evasion techniques that I identified, as well as gaps within blacklisting on mobile devices. Google followed up for details on the JavaScript cloaking, and acknowledged the gap in mobile blacklisting, which it is actively working to address. Opera requested a teleconference, and, as a result, will work with the APWG to enhance the data sources used by its anti-phishing blacklist.

**Risk to Human Users.** To ensure that my phishing websites could not harm any potential human visitors, I only distributed their URLs directly to anti-phishing entities, and the URLs had randomized paths that were infeasible to guess. In the event of form submission, my websites performed no backend processing or logging of POST data; the use of HTTPS ensured that data would not leak in transit.

**Infrastructure Usage.** I followed the terms of service of all services and APIs used for this research, and I obtained permission from Google to report URLs programmatically to Google Safe Browsing. I informed my hosting provider (Digital Ocean) about my research and obtained permission to leverage server infrastructure accordingly.

**Adverse Side-effects.** Despite my relatively large sample size for each deployment, I do not believe that the volume of URLs I reported hampered the anti-phishing ecosystem's ability to mitigate real threats. Based on the overall phishing volume per the GSB Transparency Report [47], each of my deployments accounted for less than 1% of all phishing detections made during the same period. I informed PayPal of my experiments to ensure that resources were not wasted on manually investigating my phishing URLs. I also obtained permission to use the PayPal brand and promptly disclosed the ecosystem vulnerabilities I discovered.

## 5.12 Limitations

My experimental findings should be considered alongside certain limitations. To eliminate confounding factors, I did not modify the appearance of my phishing websites between deployments, and they impersonated just a single brand (PayPal). Thus, my findings may be skewed by detection trends specific to this brand [106] and possible fingerprinting of the websites' source code. Also, my use of randomized URLs for each website may have reduced the likelihood that they would be detected. However, I believe that my approach is still realistic given attackers' re-use of phishing kits and frequent use of randomized URLs in the wild [102].

Each phishing URL I deployed (with the exception of *bit.ly* links) had a unique *.com* domain name that had not previously been reported to a blacklist. However, it was not feasible to achieve a one-to-one mapping between these domains and the pool of IP addresses available from my hosting provider. To mitigate the potential skew from IP reuse,

I distributed IP mappings as uniformly as possible within each *batch* of websites. I ultimately did not observe that URLs on certain IP addresses were more likely to be blacklisted than others.

When reporting my phishing websites, my goal was to guarantee timely discovery by the blacklists I was evaluating. Given the fast speed and near-perfect coverage I consistently observed in Experiment A, I believe that I succeeded in this goal, and thus addressed a key limitation of the original PhishFarm study [101]. Nevertheless, unlike real phishers, I did not spam real users, and I only sent reports when I initially deployed each URL. Thus, my reporting methodology may not fully reflect continuous indicators of abuse which might be observable in the wild; it may also be skewed in favor of GSB: the only blacklist to which I was able to report directly.

Finally, the scope of my experiments was limited to a subset of the different phishing website configurations that could be used by attackers. Future deployments can naturally be adapted to test other configurations which I did not consider or which might appear in the future. Although the PhishTime framework itself may fail to identify certain attacks that entirely avoid discovery, the use of additional sources of phishing URLs could address this shortcoming.

In certain experimental batches, *none* of the URLs ended up being blacklisted. It may, therefore, be more economical and more practical to use a smaller number of domains to test *different* phishing website configurations, and only increase the batch size if further scrutiny is warranted.

## 5.13 Related Work

To the best of my knowledge, the work in this chapter is the first controlled *longitudinal* empirical study of the performance of browser blacklists, and the first to propose strategies for the long-term measurement and enhancement of the protection offered by

the anti-phishing ecosystem. Although other controlled studies (e.g., which deploy new, innocuous phishing websites) have previously been done, they focused on specific anti-phishing entities and were performed over a short period.

In the PhishFarm experiments in Chapter 4, I deployed five series of 396 innocuous phishing websites, all within two weeks, to measure the effectiveness of cloaking against browser blacklists [101]. The methodology used in the PhishFarm study may seem similar to that of PhishTime, but the objectives of the two studies differ considerably. The former measured the propagation of blacklist coverage after reporting to a *single* anti-phishing entity, and, thus, showed that specific entities are vulnerable to cloaking techniques. In contrast, the longitudinal PhishTime experiments *precisely* measured blacklist speed and coverage for the ecosystem *as a whole* by simultaneously reporting to multiple entities across multiple deployments. These experiments not only showed that evasion poses a serious threat to blacklisting efficacy, but also delivered realistic long-term measurements of blacklist performance to drive specific security recommendations.

Moreover, the updated experimental design used by PhishTime more accurately represents the anti-phishing ecosystem by addressing the limitations of the PhishFarm experiments: I closely emulate the configuration of phishing websites found in the wild (e.g., by testing combinations of evasion techniques and configuring websites with HTTPS), leverage programmatic monitoring and reporting channels commonly used by automated anti-phishing systems, and analyze the persistence of blacklisting.

Peng et al. [106] followed a comparable strategy to deploy phishing websites and investigate how well VirusTotal and its sub-vendors are able to detect phishing content. This study entailed a relatively small sample size of 66 websites and consisted of a single deployment over four weeks. The study sheds light on detection models used by specific anti-phishing vendors, and performed traffic analysis similar to PhishFarm, but did not assess blacklist speed.

Other work has indirectly measured or estimated the performance of blacklists. Han et al. [51] monitored a honeypot server on which attackers installed phishing kits. In addition to measuring the ecosystem's response, the authors also monitored attackers' and victims' interactions with the kits, which provided unique insight into the execution of phishing attacks. However, because this approach hinges on attacker interactions with the honeypot, it is difficult to control variables such as the deployment time of each attack, the sample size, or the website configuration. In turn, the observations may not accurately reflect attacks within the broader ecosystem.

In Chapter 6, I analyze a large sample of phishing traffic to live phishing websites trackable through third-party web requests [103]. With this methodology, I estimate the average effect of blacklisting across the entire dataset; however, the study does not focus on evaluating the consistency of blacklists over time and thus serves as a supplement to the PhishTime findings.

Earlier studies empirically measured the blacklisting of phishing websites *after* their appearance in various feeds [72, 99, 114, 131]. Because feeds are subject to an inherent delay, the resulting measurements of blacklist speed are imprecise. However, they do provide insight into the coverage of blacklists across different platforms and the manner in which blacklists detect phishing, which inspired the general design of the PhishTime framework.

## 5.14 Summary

In this chapter, I have proposed methodology for systematically evaluating the protection provided by the anti-phishing ecosystem in the long term, with a focus on browser blacklists and phishing reporting protocols. By identifying sophisticated evasion techniques used by phishing websites in the wild and by replicating them in a controlled setting, I was able to pinpoint and help address the gaps that attackers exploit in exist-

ing defenses. With a high degree of automation, my approach provides the flexibility to deploy empirical experiments that not only realistically replicate current attacks, but can also be used to proactively test emerging threats or new mitigations.

Through analysis conducted over the course of one year, I have shown that amid a record volume of phishing attacks [47], modern blacklists are often capable of responding quickly to phishing websites. However, mobile browsers—which now account for a majority of Internet traffic—continue to receive limited blacklist protection. Also, blacklist speed and coverage both decrease considerably against evasive phishing websites commonly found in the wild. Evidence-based reporting protocols, which include data beyond a mere URL, can be an effective counter to such evasion.

Beyond making ecosystem-wide findings, the PhishTime approach can benefit the organizations that phishers impersonate. Experiments that focus on a specific brand can help measure how effectively the ecosystem can protect that brand's customers, and how well the brand implements its own anti-phishing mitigations. Given the evolution of both phishing attacks and corresponding defenses, I believe that longitudinal measurements of the ecosystem are essential for maintaining an understanding of the ecosystem's protection and evaluating new security features as they are released, such that the security of users can be continuously ensured.

In the next chapter, to show the extent to which delays in blacklisting (such as those measured by PhishTime) can impact potential victims, I apply novel methodology to precisely study the collective anatomy and effectiveness of real, large-scale phishing attacks. This study will not only revisit blacklist measurements from a different perspective but also provide deeper insight into the nature of sophisticated phishing attacks.

Chapter 6

# GOLDEN HOUR: ANALYZING THE END-TO-END LIFE CYCLE AND EFFECTIVENESS OF PHISHING ATTACKS AT SCALE

## 6.1 Introduction

Prior research has shown that traditional phishing lures, such as e-mails, have a low click-through rate (5-8%) [116] and that the likelihood that targeted users will hand over credentials to attackers is similarly low (9%) [51]. Yet, the volume of phishing attacks observed in the wild shows no signs of subsiding [5, 6]; moreover, social engineering techniques such as phishing play a central role in enabling even more harmful scams [39].

In a cat-and-mouse game, phishers collectively seek to stay one step ahead of the security community through an extensive toolkit of evasion techniques and innovations in their attacks [102], which are fueled by an underground economy [118]. In Chapters 4 and 5, I have shown that attackers' evasion techniques are capable of significantly delaying the mitigations provided by browser blacklists—a ubiquitous gatekeeper between victims and attackers [101]. However, the implications of such delays on the success of each attack are not yet well-understood, nor is the precise window of opportunity available to attackers between the launch and mitigation of their phishing websites. Actionable insight into successful phishing campaigns, victims, and the impact of interventions is therefore rare.

The challenges in measuring phishing attacks at the ecosystem level—due to the diversity of infrastructure and victims involved, and the proprietary (let alone sensitive) nature of much of the associated data—have historically accounted for this lack of insight [75, 90].

Yet, such measurements are essential to improving existing anti-phishing mitigations and developing new ones [130].

In this chapter, I present a longitudinal, end-to-end analysis of the progression of modern phishing attacks, from the time of deployment to the time of victim credential compromise. I observe that, despite their desire to remain evasive, a substantial proportion of phishing websites in the wild benefit from making requests for web resources (e.g., images or scripts) hosted by—and thus trackable by—third parties, such as the organizations being impersonated by these websites. Using this observation, I collaborate with one of the most-targeted brands in the current ecosystem [127] to develop and deploy a re-usable framework to meaningfully analyze victim traffic to live phishing websites.

From 404,628 distinct phishing URLs in the traffic dataset, I gain an understanding of the aggregate volume and timing of key events within the life cycle of phishing attacks. Next, I correlate the traffic with the original phishing e-mail lures to map the distribution phase of the attacks. Finally, I investigate the timing and success rates of attackers' monetization efforts based on the subsequent account compromise and fraudulent transactions—of the same victims—at a major financial services provider. I show that the data sampled by the approach provides *visibility* into 39.1% of all known phishing hostnames which targeted the same brand during the observation period.

I measure that the average phishing attack spans approximately 21 hours between the first and last victim visit, and that the detection of each attack by anti-phishing entities occurs an average of nearly nine hours after the first victim visit. Once a phishing attack is detected, a further seven hours elapse prior to peak mitigation by blacklists. This gap constitutes the "golden hours" which currently give attackers the opportunity to enjoy a high return-on-investment from their attacks, but which could be prevented by the ecosystem. Alarmingly, 37.73% of all victim traffic within the dataset took place *after* attack detection, and at least 7.42% of all targeted victims suffer subsequent fraud.

Moreover, my approach can identify the characteristics of particularly successful phishing attacks. I found that the top 10% largest attacks in the dataset accounted for 89.13% of targeted victims and that these attacks proved capable of effectively defeating the ecosystem's mitigations in the long term. Phishing campaigns would remain online for as long as *nine months* while tricking tens of thousands of victims in the process—using nothing more than sophisticated phishing kits on a *single* compromised domain name. As a result, I propose a practical methodology to proactively mitigate these attacks, and I deploy my approach to secure the affected victims' accounts.

My work motivates the expansion of *collaborative*, defense-in-depth anti-phishing approaches as a means to cope with phishers' evasion techniques and increasing sophistication. It underscores the importance of not only making improvements to existing ecosystem defenses such as browser blacklists, but also more widely adopting *proactive* mitigations. The contributions of this chapter are as follows:

- A longitudinal measurement study of the end-to-end life cycle of real phishing attacks representative of the modern anti-phishing ecosystem.
- A framework for the proactive detection and mitigation of phishing websites that embed external resources.
- Security recommendations to address the limitations within the current anti-phishing ecosystem based on an analysis of highly successful phishing attacks.

## 6.2 Background

As discussed in the previous chapter, the longer that phishing websites remain online and are accessible to victims, the more attackers stand to profit. Therefore, modern phishing websites seek to maximize their own longevity through a variety of strategies to remain stealthy [101].

### 6.2.1 *Measuring the Impact of Phishing*

Meaningfully assessing long-term trends in the volume of phishing attacks has historically proved to be challenging due to a lack of transparency and consistency in the methodology applied [90]. Data sources that could be effectively used for such measurements are spread throughout the ecosystem and typically held closely by their owners. Other data, such as phishing URLs, is more readily available and suitable for classification or fingerprinting purposes, but not directly coupled with attack volume or impact [24].

Since 2004, the Anti-Phishing Working Group (APWG)—an industry-wide consortium of key anti-phishing entities—has regularly published summary reports of monthly phishing volume and ecosystem attack trends based on diverse partner data [6]. Although these reports have provided phishing volume figures for over a decade, changes in methodology and data sources over time prevent longitudinal analysis and only enable conclusions such as that "phishing continues on a large scale". Research with deeper insight into the progression of phishing attacks has thus far been limited to smaller datasets or isolated scope [51, 92].

Obtaining data relating to the *damage* caused by phishing attacks (i.e., as a result of account compromise or credential theft) at specific organizations is even more challenging due to its sensitive nature in terms of both individual victims' and businesses' confidentiality. Additionally, victims themselves have shown a tendency to under-report cybercrime to authorities [37]. Aggregate summaries of such damage are thus often extrapolations based on certain assumptions [81].

## 6.3 Methodology

In this section, I discuss my approach to measuring the end-to-end life cycle of phishing websites, from the time of configuration (*B* in Figure 6.1) to the time the attack goes

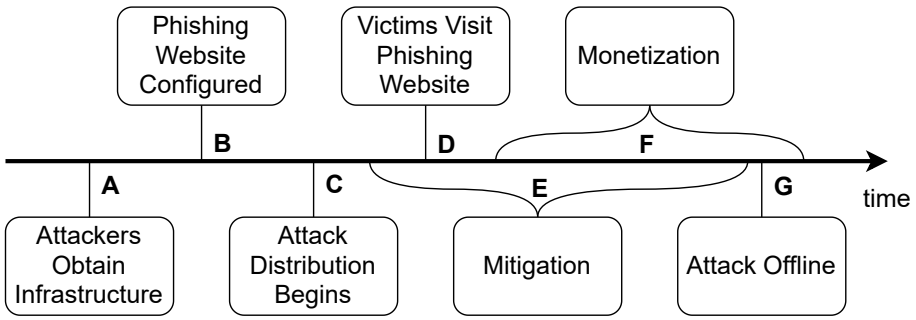



Figure 6.1: Sequence of the high-level stages of a typical phishing attack.

offline (*G*). Using the approach, in Section 6.5, I present to be what I believe the first methodical correlation of phishing attack volume with its efficacy, which includes analysis of monetization efforts (*F*). My analysis carefully dissects phishing attacks and, consequently, leads to security recommendations for existing ecosystem defenses and motivates the adoption of additional proactive anti-phishing measures. Note that malicious infrastructure configuration (*A*) is outside of the scope of the work in this chapter, as I focus purely on phishing attacks themselves [11, 14].

### *6.3.1 Phishing Attack Stages*

For users to be effectively protected against phishing, malicious websites must be mitigated in a timely manner such that each phishing attack is disrupted at the earliest possible stage, and, ideally, never shown to the user. I show an overview of the stages of a typical phishing attack in Figure 6.1, with a focus on the sequence of the stages.

Attackers first obtain infrastructure (*A*) and configure a phishing website on this infrastructure (*B*), often by installing a phishing kit. Once the website is operational, attackers begin distributing it to their victims (*C*) and victims start accessing it (*D*), as previously discussed in Chapter 2. After this point, the remaining stages are not necessarily consecutive.

Once detected by anti-phishing infrastructure, the attack will be mitigated by the ecosystem's defenses (*E*). In an optimal scenario, this mitigation would occur *before* time *D* and would prevent all future victim traffic. If these conditions are not satisfied, victim visits may continue for an extended period, and attackers will proceed to monetize the data stolen by the phishing website through various means (*F*) [126], which could entail testing stolen credentials against the corresponding platforms, or submitting fraudulent transactions using stolen financials [35, 134]. The original phishing website will eventually go offline, either as a result of take-down efforts [1] or deliberately by attackers (*G*).

### *6.3.2 Observations*

As a preliminary step in my study, in June 2018, I manually inspected a sample of live phishing websites shortly after their URLs were submitted to PhishTank [107], OpenPhish [104], or the APWG eCrime Exchange [6] (large clearinghouses of phishing URLs). I made two key observations: first, that phishing websites routinely embed resources (e.g., images, fonts, or JavaScript) hosted on third-party domains, including domains which belong to the organizations being impersonated; and second, that some phishing websites redirect the victim back to the organization's legitimate website after the victim submits his or her information.

It thus follows that third parties—including the organizations being targeted and impersonated by phishers—could, with the right methodology, directly track visitor activity on certain phishing websites by inspecting HTTP/HTTPS requests for the aforementioned web resources within their *own* systems, and by identifying referrals [40] from suspicious sources. Such tracking could capture not only victim interaction with the phishing websites, but also visits from attackers themselves as they configure and test their attacks.

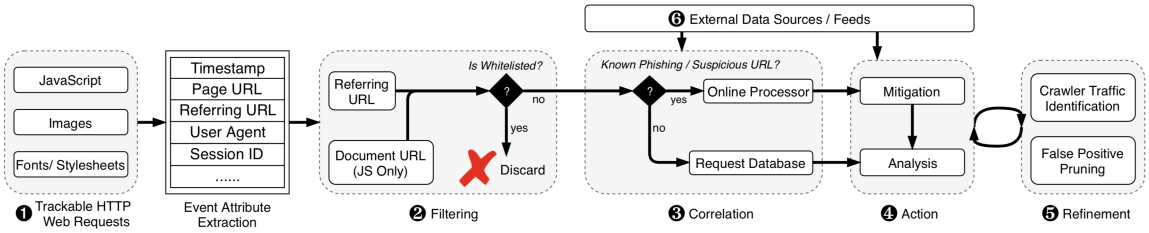



Figure 6.2: Golden Hour framework design.

Moreover, the data could be used to proactively identify phishing URLs and propagate them through the anti-phishing ecosystem. Correlating the data with victim information (e.g., if a visitor's request for a resource on a phishing website has the same session identifier as a prior visit to the organization's legitimate website [15]) could help organizations better mitigate attacks by securing any accounts tied to the victims, while simultaneously measuring the effective damage likely caused by phishing. Lastly, correlating URLs in phishing lures (e.g., e-mail messages) with victim traffic to phishing websites can paint a clear picture of the *distribution* phase of phishing attacks.

Recent work used a similar approach to identify the characteristics of successful e-mail lures and discover the corresponding URLs [130]. My analysis of web event data instead focuses on mapping the overall progression of phishing attacks: consequently, I correlate the timing of key events within phishing attacks to a deeper extent, and on a larger scale, than previous studies [51, 92]. I also consider the success of phishing attacks, and I *directly* leverage the web event data as an anti-phishing mitigation.

### *6.3.3 Data Analysis Framework*

The aforementioned analysis necessitates access to data only available to specific organizations (i.e., those commonly targeted by phishers or engaged in anti-phishing). I collaborated with one such organization—a major financial services provider—to develop and deploy a generic framework for processing the relevant data. The *Golden Hour* frame-

work, shown in Figure 6.2, extracts web tracking events associated with phishing websites for analytical purposes or as a real-time proactive mitigation.

My framework is brand-agnostic and could thus realistically be adapted for use by a broad range of organizations that have access to the appropriate data. I start by providing an abstract overview of the framework and then discuss the deployment in Section 6.4.1. In Section 6.5, I show that the framework enables insight into phishing attacks during their early *golden hours*, and that it can effectively disrupt attacks during or prior to this period.

In the *Golden Hour* framework, I first ingest web events of interest (❶), which can be obtained from raw web traffic logs (i.e., requests for images or style elements) or pre-processed data from web trackers or JavaScript web application code. I annotate each ingested event with a timestamp and extract further attributes, such as the IP address, user agent, session identifiers (i.e., from prior requests), referring URL, and the main page URL which was visited. I then take the latter two URL attributes and apply whitelist filtering (❷) to eliminate benign events which would normally be expected to be seen in this context, such as requests to the organization's legitimate website or requests with referrers on approved partner websites. Thereafter, I correlate (by substring matching) the URLs of the remaining events with a recent list of known phishing URLs from additional data sources (❸); this correlation enables the discovery of new phishing URLs which might only share a similar hostname or path with a previously-reported URL, but differ otherwise. It is also possible to apply phishing URL classification heuristics to identify previously-unknown URLs of interest [41].

The event correlation can take place in an online manner, or be deferred, in which case events are archived for later analysis (❹). In both cases, to allow for scalability, a chosen *observation window* defines a range of time (i.e., before and after a URL is reported) within which correlations for a given phishing URL are made. Successive reports of the same URL

| | Date Range | No. of Samples |
|---|---|---|
| **Golden Hour web events** | 10/01/18 - 09/30/19 | 22,553,707 |
| **(distinct phishing URLs)** | | 404,628 |
| **E-mail reports** | 09/01/19 - 09/30/19 | 68,502 |
| **APWG phishing URLs** | 10/01/18 - 09/30/19 | 52,116 |
| **Organization's phishing URLs** | 10/01/18 - 09/30/19 | 37,438 |
| **Fraudulent account transactions** | 10/01/18 - 09/30/19 | Not disclosed |
| **Compromised user accounts** | | |

Table 6.1: Overview of the datasets analyzed.

naturally extend the observation window; otherwise, correlations against unnecessary data can be avoided.

Events that are identified as phishing are additionally marked for immediate mitigation. Over time, I further refine the archived events (❺) by identifying false positive correlations, noise from automated (i.e., web crawler) traffic, and phishing URLs detected at a later point in time, with the use of statistical analysis and external data sources (❻).

To benefit from my framework's mitigation capabilities, it should ideally be deployed *online*, on a stream of live (or recent) data during the ingestion stage (❶). However, the framework can also process archived (i.e., historical) event data alone. In the long term, as the anti-phishing ecosystem builds ground truth (i.e., by having access to a vetted list of known phishing URLs), both approaches will enable the same level of analytical insight. Thus, the framework can accommodate different data ingestion strategies to support the infrastructure of the organization deploying it.

## 6.4 Dataset Overview

I deployed the *Golden Hour* framework to collect and analyze one year of phishing web traffic data between October 1, 2018, and September 30, 2019 (inclusive), at the same organization mentioned in the previous section—a major financial services provider and one of the most-targeted brands within the current ecosystem [26, 86]. I provide an overview of the scope of all of my datasets in Table 6.1. Note that this data was collected ethically and in compliance with user privacy laws within the originally-intended context (see Section 6.8.3).

### *6.4.1 Data Collection*

I operated the *Golden Hour* framework in an *online* manner from July 1, 2019, through September 30, 2019, and additionally processed archived data from the preceding nine months. To efficiently query a data warehouse, I limited my *observation window* for web event data (as discussed in Section 6.3.3) to one week before and one week after the corresponding hostname appeared in a phishing feed. I found that this approach did not lead to the omission of any relevant events, as phishing URLs which remained live for longer periods would reappear in the feeds at a later date, and would thus also be extracted by my framework for analysis.

The resulting dataset initially contained a total of 22,553,707 web events representative of traffic to phishing websites from victims, attackers, and security crawlers alike. Using the traffic data, it is possible to gain detailed insight into stages *B*, *D*, *E*, and *G* within the life cycle of phishing attacks. For the framework's correlation (❸) and refinement (❺) steps, I programmatically queried additional data sources: phishing URLs for the same brand in the APWG eCrime Exchange feed, the organization's proprietary phishing URL

feed, and the organization's proprietary automated (i.e., crawler) traffic detection system (❻).

During my deployment, I pruned 3,194,031 events by identifying traffic to legitimate websites (based on a whitelist and manual review) and false-positive URLs that were under-represented in the phishing feeds or flagged as such by the organization. Thus, my final dataset contained 19,359,676 total events. These events corresponded to 404,628 distinct phishing URLs—more than either phishing feed I considered, as my hostname correlation process enables the identification of unreported variants of URLs similar to those which appeared in feeds. However, additional types of data are required to obtain timings of attack distribution (stage *C*) and monetization (stage *F*) and thus complete my end-to-end analysis, as these are not captured by the traffic dataset alone.

**Phishing URL Distribution:** To measure URL distribution to victims, I extracted metadata from phishing e-mails that users forwarded (i.e., as spam reports) to the organization. The timestamps within the original e-mail lures allow us to calculate when phishers originally distributed their attacks. To correlate these timestamps with web events in my traffic dataset, I extracted URLs from each e-mail and followed redirects (if any) to obtain the URL of the final phishing landing page. In cases when a redirect was followed, or if the phishing URL was no longer accessible, I would additionally query the organization's internal anti-phishing system to obtain any other landing page URLs known to be previously associated with the URL in the e-mail. To complete the correlation, I search for events within the traffic dataset with the same hostname and a common path.

I was able to correlate 21,244 e-mail reports with phishing URLs in my event dataset[1]. I found that 84.44% of these e-mails contained a timestamp detailed enough (i.e., date, time, and timezone) for my analysis. Determining final landing page URLs from links within

[1]The uncorrelated e-mails either were outside of the *visibility* of my approach, or had redirection chains that could not be reconstructed. I discuss the relatively small size of my e-mail dataset in Section 6.9.

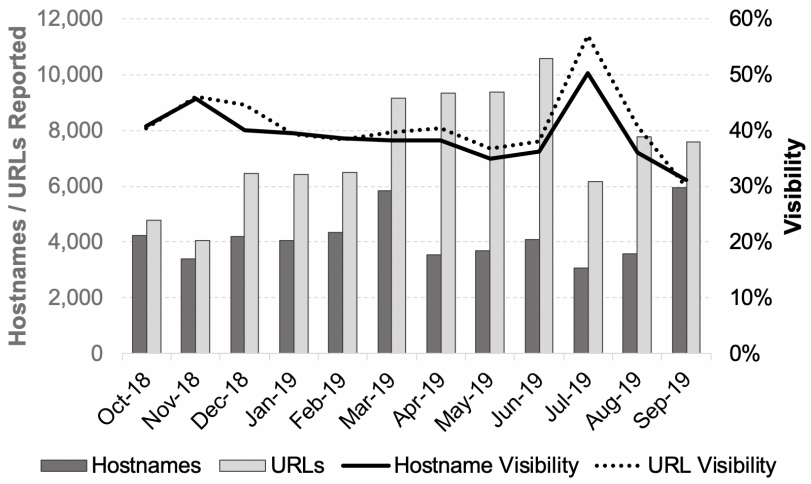



Figure 6.3: Visibility of phishing websites in the Golden Hour dataset.

the e-mail proved integral, as only 3.99% of e-mails contained the same URL as the final phishing page (i.e., others made use of redirection).

**Account Compromise and Monetization:** To understand one way in which criminals exploit credentials from phishing victims, I analyzed session identifiers programmatically extracted from events in the traffic dataset (i.e., victim visits to a phishing website which had cookies from a prior interaction with the legitimate organization's website). The organization then mapped these identifiers to user accounts and provided timestamps of fraudulent transactions associated with these accounts, and timestamps of when corresponding credentials appeared in a public dump. I could then correlate these timestamps with the victims' original interaction with the phishing page per the traffic dataset.

Due to the sensitive nature of user information, I present my findings related to this data in aggregate form only (in Section 6.5). Note that no Personally Identifiable Information (PII) was given to me for the purposes of this analysis.

### 6.4.2 Level of Visibility

An immediate question that arises about my analysis concerns the level of *visibility* than can be achieved by the *Golden Hour* framework. I define *visibility* as the proportion of the total population of phishing websites which can be analyzed through my approach. While I cannot provide a definite visibility measurement, as this would require knowledge of *all* phishing campaigns that target the organization, I estimate the visibility of my dataset by dividing the number of distinct hostnames with at least one associated web event by the total number of phishing hostnames, for the same brand, known from other data sources during the data collection period (i.e., the URLs reported to the APWG or found in the organization's phishing URL feed). I also calculate the same ratio for full URLs. Note that it is easier for phishers to create multiple paths on a single domain compared to multiple subdomains; thus, the hostname ratio better represents unique attacks.

I found that my approach had visibility into an average of 39.1% of all hostnames and 40.9% of all URLs which were found in the aforementioned feeds (and, thus, targeted the organization) during the data collection period. I present the visibility ratios by month in Figure 6.3.

Given the evasiveness of modern phishing attacks, I suspect that the phishing URL feeds underestimate the population of all phishing URLs found in the wild [102, 125]. Consequently, however, the same would apply to the set of URLs for which I collected event data through *Golden Hour*. Therefore, I believe that my assessment of visibility is realistic.

The degree of visibility for both hostnames and URLs remained fairly consistent throughout the data collection period, with the exception of July 2019. During this month, I observed a spike to 50.2% and 57.1% visibility, respectively, which coincided with the launch

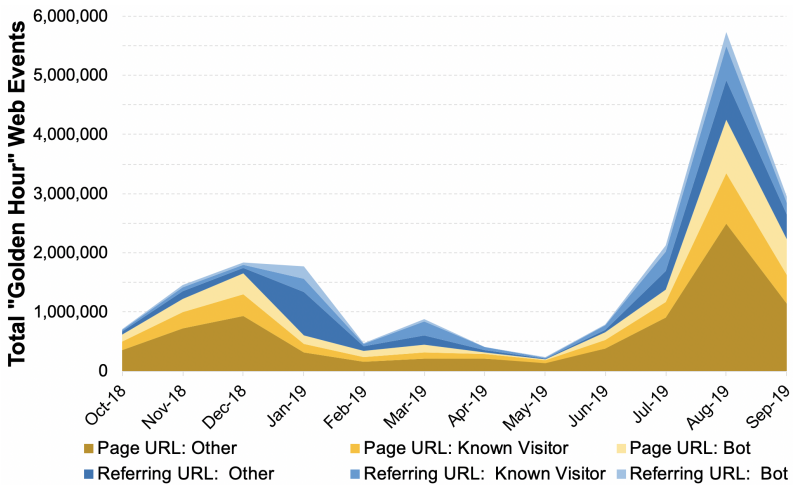



Figure 6.4: Distribution of Golden Hour web events by month.

of numerous sophisticated, large-scale attacks that were detectable by my approach. I discuss these attacks in more detail in Section 6.7.

Per the APWG eCrime Exchange, the brand in my dataset accounted for 10.6% of all phishing hostnames (with known brands) during the same period. This allows extrapolation of the possible visibility of my approach into the population of phishing websites.

### *6.4.3 Event Distribution*

I collected web events of two broad types: visits that occurred directly on phishing websites (*Page URLs*) and referral traffic from a phishing website back to the organization's legitimate website (*Referring URLs*). I show the monthly distribution of these events in my dataset in Figure 6.4. I observe that phishing attacks are not uniformly distributed; some months see substantially more traffic than others. Historically, phishing attacks have been associated with a certain seasonality, particularly near holidays. The spike in the final three months of my dataset is consistent with the Q3 2019 APWG report, which found this period to have the largest volume of phishing URLs in three years [7]. However, I

| | Known Visitor | Crawler | Other | **Total** |
|---|---|---|---|---|
| **Page URL** | 2,968,735 | 2,934,976 | 7,982,475 | 13,886,186 (71.73%) |
| **Referring URL** | 1,879,179 | 820,716 | 2,773,595 | 5,473,490 (28.27%) |
| **Total** | 4,847,914 (25.04%) | 3,755,692 (19.40%) | 10,765,070 (55.56%) | 19,359,676 |

Table 6.2: Breakdown of Golden Hour web events by type.

expose a limitation of counting URLs alone as a measurement of overall phishing volume, as the spike in my traffic dataset is far more dramatic than the change in total URLs[2].

In Table 6.2, I further subdivide the events by the type of user. Events from *Known Visitors* are those which contain a session or device identifier that was previously known to the organization, and can thus be linked with certainty to a known account at the organization. *Crawler* events are those which I or the organization classified as automated traffic based on request attributes. The *Other* events fall into neither category but follow a similar distribution to *Known Visitors*, and thus represent potential victims which cannot be immediately traced back to an account at the organization.

To ensure consistency across my measurements in the following sections, I define the set of *Compromised Visitors* as those *Known Visitors* whose accounts were subsequently either accessed by an attacker or had at least one fraudulent transaction. I consider only these events in my analysis of monetization efforts, as the sequence of observations strongly suggests that a phishing attack succeeded against the corresponding victims. I do

[2]I believe that both of these spikes are associated with the effectiveness (and proliferation) of highly sophisticated phishing websites, which I characterize in Section 6.7.1.

not disclose the total number of unique victims within these two sets for reasons discussed in Section 6.9.

## 6.5 Progression of Phishing Attacks

To create an end-to-end timeline of the progression of phishing attacks, I calculate the relative difference between the timestamp of each *Golden Hour* web event and the original detection of the corresponding phishing URL within a feed, as correlated by my framework. I calculate similar timestamp differences for e-mail lures, account compromise, and fraudulent account transactions. I can then plot a histogram of victim traffic relative to attack detection, alongside the average timestamps of key attack milestones. Note that the effect of outliers on these averages is inherently suppressed by my use of a fixed *observation window* for each phishing URL's web events.

In Figure 6.5, I show such a histogram for *Compromised Visitors*: in other words, every user represented in the figure was highly likely to have been successfully fooled by a phishing attack. I count multiple events from the same victim on the same phishing website only once. For brevity, I do not separate *Page URL* and *Referring URL* events in my figures, as these did not differ significantly except in the success rates of subsequent account compromise (discussed in Section 6.5.3).

I observe that phishers enjoy a large window of opportunity when carrying out their attacks. Nearly nine hours elapse between the average first victim visit and detection by the ecosystem. By this time, the phishing websites have already lured 62.73% of victims. Moreover, victim visits continue at a slower pace for the next 12 hours. I show the Cumulative Distribution Function (CDF) of *Compromised Visitor* web events in Figure 6.6a. Despite the 21-hour time frame (-08:44 to +12:26) of a typical phishing attack illustrated in Figure 6.5, there exist some attacks with a longer overall duration.

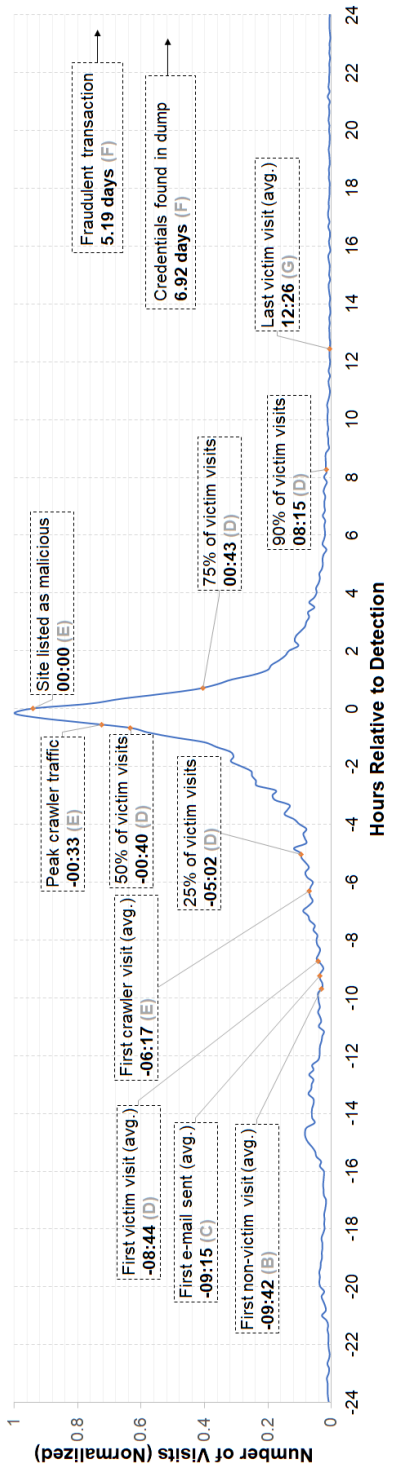
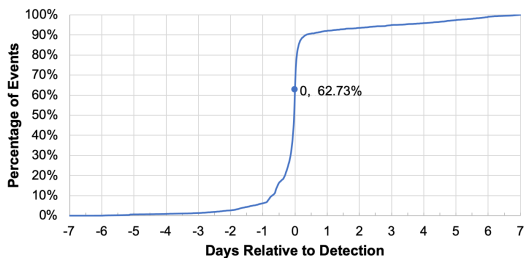
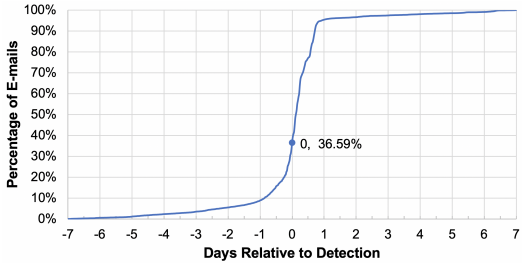


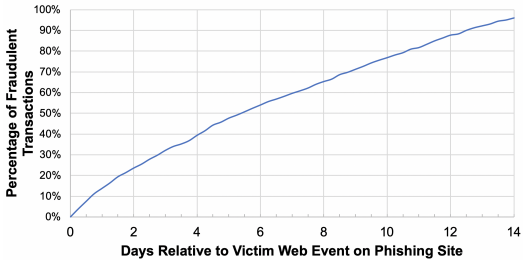

Figure 6.5: Histogram of Compromised Visitor traffic to phishing websites, annotated with different attack stages.




(a) Compromised Visitor traffic.




(b) Phishing e-mail distribution.




(c) Fraudulent transactions.

Figure 6.6: Cumulative Distribution Function (CDF) plots depicting key phishing attack stages.

### *6.5.1 Initial Traffic*

The average first non-victim visit to each phishing website occurs 9 hours and 42 minutes prior to attack detection, as shown in Figure 6.5. I believe that such visits are representative of attackers' initial testing of each phishing website.

I performed an unequal variance T-test [113] to compare the distribution of the relative timestamps of the *first* event (for each attack) within the *Other* category to the *first* event for *Known Visitors*. I find the means of the two distributions to be statistically significantly different, with a $p$ value of 0.011. Furthermore, in Table 6.3, I show that top geolocations within the former set closely coincide with countries disproportionately associated with cybercrime [67] (and inconsistent with the organization's customer base).

| Country | Other Traffic | Country | Known Visitor Traffic |
|---|---|---|---|
| United States | 32.84% | United States | 65.48% |
| Morocco | 9.17% | United Kingdom | 6.15% |
| Indonesia | 8.16% | Canada | 4.26% |
| United Kingdom | 6.08% | Italy | 3.05% |
| Algeria | 3.73% | Spain | 2.78% |
| Canada | 2.99% | Australia | 2.58% |
| Germany | 2.88% | Germany | 2.29% |
| Brazil | 2.35% | Mexico | 1.46% |
| Tunisia | 2.29% | France | 0.93% |
| Italy | 2.24% | Netherlands | 0.79% |
| France | 1.92% | Brazil | 0.72% |
| Iraq | 1.60% | Singapore | 0.64% |
| Egypt | 1.44% | Ireland | 0.40% |
| Spain | 1.39% | Belgium | 0.40% |
| Nigeria | 1.39% | Portugal | 0.38% |

Table 6.3: Geolocation of initial visits to phishing sites, by traffic category.

### *6.5.2 Phishing E-mail Distribution*

I show the CDF of phishing e-mail distribution in Figure 6.6b. Note that prior to attack detection, the cumulative proportion of victim visits to phishing websites (in Figure 6.6a) grows at a faster rate than e-mails sent. In other words, traffic from phishing e-mails to phishing websites drops after attack detection, as should be expected following the intervention of spam filters. However, just one day after detection, the rate of victim visits once again starts outpacing the sending of e-mails. This suggests that victims will follow non-blacklisted links in old e-mails, and, thus, attackers continue to profit without

further intervention. We, therefore, believe that effective take-down remains an important secondary mitigation to suppress long-lasting phishing attacks [1].

### *6.5.3 Progression of Monetization Efforts*

In my dataset, the accounts of 7.42% of distinct *Known Visitors* subsequently suffered a fraudulent transaction; I believe this represents a lower bound on success rates and subsequent damage from phishing, as my approach does not identify victimization of the *Other* traffic. After each victim's visit to a phishing website, I found that such a transaction would occur with an average delay of 5.19 days. However, as I show in Figure 6.6c, fraudulent transactions grow consistently over a 14-day period, with the earliest ones occurring less than one hour after a victim visit. Although 3.99% of fraudulent transactions in my dataset occurred *after* the 14-day period, the increasing potential for mitigation encourages attackers to act quickly.

The credentials of 63.61% of these compromised victims would additionally appear in a public dump, with an average delay of 6.92 days. This trend suggests that criminals tend to first monetize the accounts of their victims, and only later sell credentials within underground economies [16].

The Golden Hour dataset does not provide insight into the monetization of each victim's stolen personal information beyond the organization's own systems. I find that the average victim makes 2.43 page loads during his or her interaction with a phishing website—enough to visit a landing page and submit credentials. Some victims, however, made substantially more visits during a single session. After inspecting the chain of phishing URLs visited in such sessions, I believe that such victims provide additional personal information to the phishing website (i.e., one with multiple data collection forms), and could thus suffer from identity theft or other financial fraud. Per my dataset, I observed that victims with an above-average number of page loads who also appeared in a *Refer-*

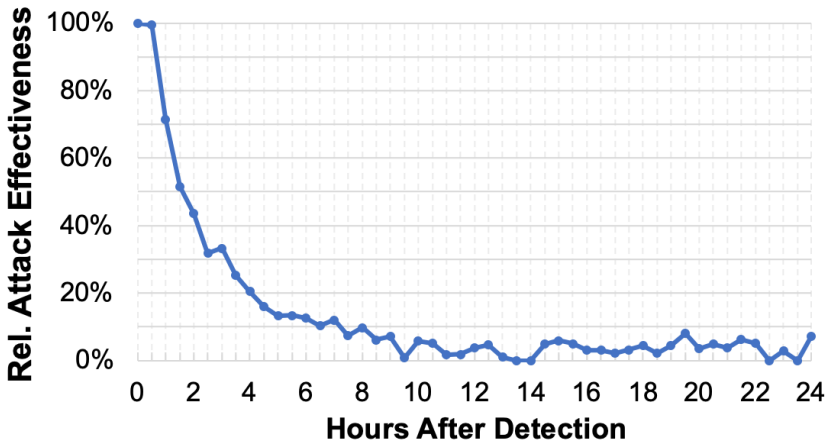



Figure 6.7: Impact of blacklisting on phishing effectiveness.

*ring URL* event (i.e., returned to the organization's website after presumably completing interaction with the phishing website) were 10.03 times more likely to later encounter a fraudulent withdrawal from their account.

### *6.5.4 Estimating Blacklist Effectiveness*

Given the ubiquity of browser blacklists, the role of blacklisting in preventing phishing in the wild is a key measurement I seek to estimate. The mitigation from blacklists can be delayed for two main reasons: failure of backend systems to flag a given phishing URL or the lag between backend flagging and data propagation to clients (e.g., browsers) [48]. This lag period may vary between the same browser on different devices due to differences in cache state [101].

We can meaningfully estimate the overall impact of blacklisting on phishing attack effectiveness by calculating the ratio of *Compromised Visitors* for browsers with native blacklists and *Compromised Visitors* for *all* browsers, at regular time intervals after attack detection (i.e., after the midpoint of Figure 6.5), and subsequently comparing this ratio to a baseline ratio just prior to detection. This ratio is not sensitive to the decrease in absolute phishing traffic as it simply isolates the likelihood that the phishing attack will be suc-

cessful (*Compromised Visitors* are visitors who likely submitted credentials to a phishing website).

As I show in Figure 6.7, blacklisting starts noticeably reducing the relative effectiveness of phishing attacks within one hour after detection, at which point the ratio of *Compromised Visitors* drops to 71.51%. By the end of the second hour, the ratio drops further to 43.55%: at this point, attacks are less than half as effective as they were originally. The effectiveness continues declining more slowly until the seventh hour and thereafter stabilizes within the 0-10% range.

Blacklists are clearly an effective mitigation overall, but attackers can and do abuse delays in blacklists' reaction, as I have demonstrated. In addition, certain evasion techniques, which I discuss in Section 6.7.1, can prevent blacklisting from occurring, even after attack detection. Additional mitigations are required to thwart the trickle of *Compromised Visitor* visits which I observed many hours after detection.

By comparing the mitigation provided by blacklists in Figure 6.7 with the victim traffic volume in Figure 6.5, we can model the potential reduction in phishing victims that would occur if blacklisting performance were improved by the ecosystem, or if attack *discovery* occurred earlier. Given the observation that a large proportion of victim traffic occurs within a short time window, even a small increase in the speed of blacklisting (or other browser-based mitigations) could drastically reduce the potential number of victims who end up visiting phishing pages. For example, in Figure 6.5, 25% of all victim visits happened in just 83 minutes (during the interval from 40 minutes before detection to 43 minutes after detection). Faster blacklisting could help reduce the number of victims accordingly.

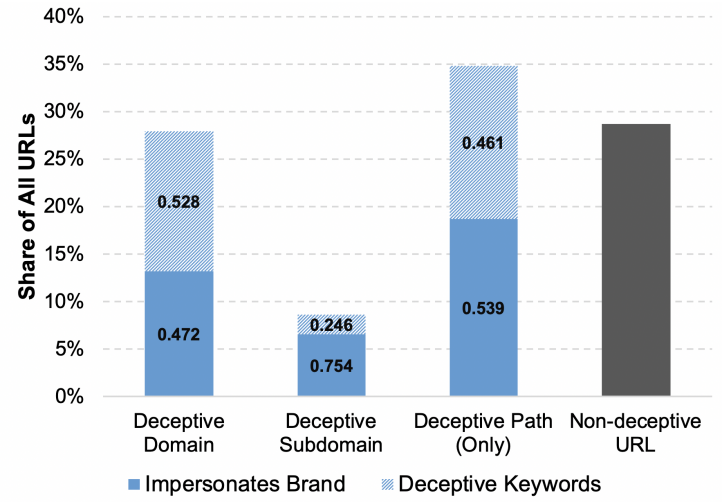



Figure 6.8: Classification of phishing URL content in the Golden Hour dataset.

## 6.6 Phishing Website Characteristics

In this section, I analyze metadata related to the phishing websites in my dataset in an effort to better understand the characteristics of successful attacks. I consider all phishing URLs with at least one *Compromised Visitor* event.

### *6.6.1 Phishing URL Classification*

Attackers have traditionally crafted phishing URLs to deceive victims by either mimicking the brand being impersonated by the phishing website (e.g., *www.brand-alerts.com*), or by including misleading keywords which convey a desire to help the victim (e.g., *secure-my-account.com*) [41].

I apply the classification scheme from Table 3.3 (in Chapter 3) to the URLs in my dataset and show the results in Figure 6.8 [102]. I observe that 28.70% of all URLs have no deceptive content whatsoever; 34.76% have non-deceptive domains with deceptive paths only. 8.64% use deceptive subdomains on a non-deceptive domain, and the remaining 27.90% have deceptive domains (0.52% with Punycode [21]). The nature of deceptiveness

is similarly split between brand names and misleading keywords, except in the case of subdomains, which favor brand names. Bare IP addresses were negligible in my dataset and thus are excluded from the figure.

The vast majority of phishing URLs (98.58%) were hosted on traditional, paid domain names. Only 0.79% of URLs leveraged subdomains from free hosting providers; 0.63% had domains with free ccTLDs [102]. However, compromised hosting infrastructure plays a key role, which I assess in Section 6.7.

With the increasing use of mobile devices to browse the Internet, the importance of URL content has diminished (i.e., because of limited screen real estate on such devices) [73]. However, the heavy use of redirection in phishing lures allows attackers to somewhat comfortably continue using deceptive URLs (which would otherwise be easily detectable by text-based classifiers) on their landing pages.

### *6.6.2 Device and Browser Type*

| **Browser Name** | **Traffic Share** |
|---|---|
| Chrome Mobile | 29.72% |
| Safari Mobile | 22.38% |
| Chrome | 21.56% |
| Samsung Browser | 7.97% |
| Edge | 5.53% |
| Safari | 4.10% |
| Firefox | 3.66% |
| Internet Explorer | 3.21% |
| Other | 1.87% |

(a) By browser.

| **Device** | **Traffic Share** |
|---|---|
| Android | 35.70% |
| Windows | 28.13% |
| iOS | 27.03% |
| OS X | 8.35% |
| Other | 0.79% |

(b) By device.

Table 6.4: Known Visitor traffic share by browser and device.

As shown in Table 6.4, mobile devices accounted for 62.73% of all victim traffic in my dataset. Browsers protected by *Google Safe Browsing—Chrome*, *Safari*, and *Firefox*—accounted for 81.42% of the traffic (roughly consistent with their overall market share) [120]. The wide use of these browsers, in particular on mobile platforms, underscores the importance of the efficacy of the anti-phishing features which they natively include.

The *Samsung Browser*, which does not currently include safe browsing functionality to the best of my knowledge, and thus leaves users particularly vulnerable to phishing, had a disproportionate representation of 7.97% in my dataset.

I studied the behavior of individual browsers in detail in Chapters 4 and 5, and such behavior is, therefore, outside of the scope of my analysis in this section [101]. However, I did observe that browsers with proactive heuristic-based anti-phishing features (*Edge* and *Internet Explorer*) led to relatively fewer *Compromised Visitor* visits in the long term compared to other browsers with only blacklists.

### *6.6.3 Use of HTTPS*

The web has moved away from traditional HTTP in favor of encrypted communication over HTTPS; phishers started following this trend in 2017 [5], which has been simplified through the wide availability of free SSL certificates [69]. Within my entire dataset, 66.85% of distinct URLs used HTTPS. However, these URLs accounted for 85.77% of the *Compromised Visitors*. Phishing attacks with HTTPS thus proved about three times more successful than HTTP. Even though some successful phishing attacks still occur on unencrypted websites, I believe that such infrastructure is now of reduced interest to phishers. Simultaneously, the potential impact of Certificate Authorities (CAs) in helping prevent abuse—especially on attacker-controlled domains—has grown. For example, CAs could more closely examine requests for certificates for suspicious-looking domains or domains with a malicious history, or proactively warn organizations of potential attack websites.

## 6.7 Phishing Campaign Longevity

| Rank | First Seen Date | Last Seen Date | Campaign Duration (Days) | Known Visitor Events | Average Events Per Day | Distinct URLs Reported | URL Text Classification | Domain Type |
|---|---|---|---|---|---|---|---|---|
| 1 | 01/06/2019 | 09/22/2019 | 259 | 145,306 | 560 | 41 | Deceptive Path Only | Compromised |
| 2 | 08/30/2019 | 09/26/2019 | 27 | 115,616 | 4,329 | 41 | Deceptive Subdomain | Compromised |
| 3 | 07/20/2019 | 09/14/2019 | 56 | 102,601 | 1,847 | 40 | Non-deceptive | Free Subdomain |
| 4 | 01/11/2019 | 01/15/2019 | 4 | 82,636 | 20,487 | 6 | Deceptive Path Only | Regular Registration |
| 5 | 06/14/2019 | 06/20/2019 | 6 | 71,478 | 11,681 | 56 | Non-deceptive | Compromised |
| 6 | 04/21/2019 | 05/27/2019 | 36 | 71,037 | 1,992 | 39 | Deceptive Path Only | Regular Registration |
| 7 | 08/11/2019 | 08/17/2019 | 5 | 59,911 | 11,296 | 40 | Deceptive Subdomain | Free Domain |
| 8 | 03/14/2019 | 04/22/2019 | 39 | 55,147 | 1,427 | 81 | Deceptive Subdomain | Regular Registration |
| 9 | 08/30/2019 | 09/26/2019 | 27 | 50,402 | 1,877 | 28 | Deceptive Subdomain | Compromised |
| 10 | 01/07/2019 | 01/07/2019 | 1 | 49,627 | 49,627 | 8 | Deceptive Subdomain | Free Subdomain |
| 11 | 12/22/2018 | 12/26/2018 | 4 | 44,502 | 10,806 | 45 | Non-deceptive | Compromised |
| 12 | 06/23/2019 | 06/28/2019 | 6 | 42,574 | 7,708 | 22 | Deceptive Subdomain | Free Subdomain |
| 13 | 09/24/2019 | 09/25/2019 | 2 | 42,406 | 21,203 | 29 | Deceptive Domain | Regular Registration |
| 14 | 12/12/2018 | 01/02/2019 | 21 | 38,484 | 1,814 | 16 | Deceptive Path Only | Compromised |
| 15 | 10/06/2018 | 02/22/2019 | 140 | 32,591 | 233 | 39 | Deceptive Path Only | Compromised |
| 16 | 12/11/2018 | 12/29/2018 | 18 | 30,983 | 1,768 | 63 | Deceptive Subdomain | Regular Registration |
| 17 | 10/31/2018 | 03/24/2019 | 145 | 30,853 | 213 | 90 | Deceptive Path Only | Regular Registration |
| 18 | 09/12/2019 | 09/22/2019 | 10 | 30,781 | 2,990 | 23 | Deceptive Path Only | Compromised |
| 19 | 03/19/2019 | 03/24/2019 | 4 | 23,552 | 5,399 | 21 | Deceptive Path Only | Regular Registration |
| 20 | 08/13/2019 | 08/15/2019 | 3 | 22,254 | 7,418 | 16 | Deceptive Domain | Regular Registration |

Table 6.5: Top phishing campaigns by the number of Known Visitor events.

Prior research has stipulated that individual phishing attacks tend to be short-lived and that they capitalize on the narrow gap between deployment and detection [76]. Despite some caveats, I have made a similar observation in Section 6.5. However, these observations do not capture trends within broader phishing campaigns, which may entail a group of organized criminals involved in the successive deployment of persistent and sophisticated attacks.

To gain better insight into long-term phishing campaigns, I group phishing URLs from events in my dataset by domain (or by hostname in the case of free subdomain hosting providers). I then sort the groups by the total number of unique *Known Visitor* events to capture variations in hostname or path for attacks that are likely related[3]. I define the *date range* of a campaign as the time between the first and last web event from a *Compromised Visitor*; I found the average *date range* to be 13.55 days.

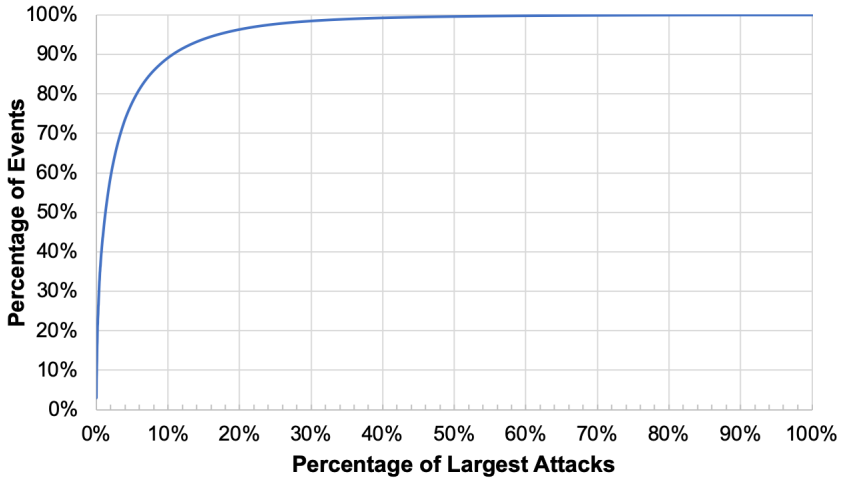



Figure 6.9: Share of Known Visitor events by top attacks.

I discovered that the top 5% of attacks accounted for 77.79% of *Known Visitor* events within my dataset, and the top 10% for 89.13% (as reflected in the CDF in Figure 6.9). I then manually analyzed the top 20 campaigns (these alone accounted for 23.57% for *Known Visitor* events), some of which lasted several months each, as shown in Table 6.5. I also determined whether they were hosted on compromised domains (i.e., otherwise belonging to a legitimate website) or domains directly controlled by attackers.

[3]Some threat actors pivot across different infrastructure and might thus be underestimated by domain-based grouping of attacks. Confidently grouping attacks by other attributes, such as a phishing kit signature or drop e-mail, would require additional data. The same applies in case different threat actors were to leverage a single domain.

### 6.7.1 Sophistication and Evasion

To understand the success of the top phishing attacks, I manually inspected the content (and, when possible, phishing kits) of high-impact phishing URLs that were live during the *online* deployment between July 1 and September 30, 2019. I identified such URLs by spikes in the number of *Known Visitor* events associated with any individual hostname.

The characteristics I found contribute to the attacks' success not only by avoiding detection by anti-phishing infrastructure, but also by more effectively targeting human victims. I quantify my observations to the extent possible given my methodology; however, a more comprehensive measurement would be suitable for future work.

**Broad Data Collection:** The sophisticated phishing websites which I analyzed mark a clear departure away from single-page login forms, and thus venture far beyond mere theft of usernames and passwords [22]. Phishers fully match the page structure of the victim organization's website, complete with a homepage with links to (fraudulent) login pages and resources in case the victim was to navigate away from the initial landing page. Once the victim returns to the login page and starts interacting with the phishing website, it will seek to harvest extensive personal and financial information, identity documents, and even photographs (i.e., *selfies*) to steal and more effectively monetize victims' identities.

**Automatic Translation:** Five of the phishing websites in Table 6.5 used the visitor's geolocation to automatically translate their deceptive content. Manual analysis of the phishing kit used on one of these websites revealed a total of 14 language options that coincided with the targeted brand's major markets.

**Human Verification:** I observed that as part of a URL redirection chain, some attackers would show a reCAPTCHA challenge [133] prior to redirecting the victim to the phishing landing page. Also, one specific phishing kit showed a CAPTCHA challenge directly on its landing page before allowing the victim to input any credentials. Such

challenges not only hamper the verification of phishing content by automated systems, but may also trick users into proceeding due to the use of CAPTCHA on legitimate websites.

**Cloaking:** All the phishing websites which I analyzed leveraged server-side cloaking, a well-known technique that seeks to block traffic based on a blacklist (or a whitelist) of request attributes such as IP address or hostname, as discussed in Chapter 3 [58, 102, 125]. Such cloaking intends to restrict access from security crawlers or other non-victim entities. Also, some phishing kits include an initial landing page that contains nothing but a simple piece of JavaScript code or an HTML Meta tag to redirect the victim to the true phishing page. Such code could defeat basic crawlers that look at static HTML only.

**Victim-specific Paths:** Eight of the campaigns in Table 6.5 had a landing page that automatically generates a sub-path unique to each visitor's IP address, and then immediately redirects to that path. The path is not visible to other IP addresses, and would thus evade crawlers visiting a previously-generated path rather than the attack's initial landing page.

**Fake Suspension Notices:** As a deterrent to take-down efforts [1], when a visitor fails *cloaking* checks, I observed that several phishing websites displayed a misleading message indicating that the domain has been suspended, rather than a generic HTTP 404 or 403 error message [40]. This could lead hosting providers to believe that a malicious page had already been taken down, and the provider would take no further action, when in reality the website was still capable of serving phishing content.

**Man-in-the-Middle Proxies:** Rather than a traditional phishing kit, two of the large phishing attacks that I analyzed used a proxy that would make live requests to the legitimate organization's website and display the page to the user while intercepting all data submitted [36]. Such proxies can defeat most forms of two-factor authentication [129] and may require special care to mitigate by the targeted organization.

### 6.7.2 Attack Mitigation

While analyzing the sophisticated attacks in Table 6.5, I simultaneously manually reported them to anti-phishing entities and hosting providers. By the time the many original URLs were blacklisted, the attackers would have redeployed subsequent attacks on different subdomains or paths, which would, in turn, necessitate another cycle of blacklisting. In this manner, attackers can stay one step ahead of the ecosystem. When paired with bulletproof hosting (i.e., resistant to take-down from abuse reports) [65] or successive re-compromise of legitimate, albeit vulnerable, infrastructure, such attacks remain effective for prolonged periods.

To help overcome the challenges faced by the ecosystem, I adapted the *Golden Hour* framework to perform proactive mitigation of attacks. I reported events corresponding to *Known Visitor* back to the victim organization, such that the organization could flag accounts to prevent successful compromise or re-secure accounts that had already been compromised. I reported tens of thousands of distinct events in this manner, which has motivated the permanent adoption of my framework by the organization.

The *Golden Hour* framework can also be used to discover previously-unknown phishing URLs based on heuristics such as textual URL content (applied during correlation) or context. Such URLs can then be reported to blacklists and propagated through the ecosystem. Due to technical limitations, I did not automate this aspect of the framework during the deployment. In a retrospective analysis, I found that this would have potentially increased the number of web events in the dataset by 7.28%, which, if reported, could help narrow the gap in the detection of phishing attacks by the ecosystem.

## 6.8 Discussion

Although individual evasion techniques might not suffice to defeat the modern anti-phishing ecosystem, the increased degree of sophistication which arises from the combination of such techniques poses a key threat. I have shown that in terms of the number of victims compromised, sophisticated and persistent phishing attacks dominate, and should, therefore, be a priority for the ecosystem. At a more granular level, both the response time of blacklists (which protect victims once a phishing attack is detected) and speed of initial detection (which closes attackers' window of opportunity), through spam filters and other means, warrant improvement.

### *6.8.1 Data Sharing*

The mere fact that so many of phishing websites in my dataset embed third-party resources shows that attackers do not fear being detected by certain organizations. Consequently, there is an opportunity for increased data sharing across the ecosystem to better detect threats based on proactive intelligence indicative of attacks: the web events from *Golden Hour* are just one example of such intelligence.

**Reporting Phishing:** Sophisticated phishing attacks thrive from being seemingly untouchable by the ecosystem. In the case of cloaked phishing websites, simple URL-based reporting to blacklists—such as what is currently commonly carried out through automated systems and web submission forms [48]—fails to provide sufficient context for the blacklist to expeditiously and effectively verify the phishing content.

With only a URL in hand, the blacklist may not be able to determine the parameters required to defeat the cloaking. I experienced this phenomenon when manually reporting certain URLs from the sophisticated attacks in Table 6.5 to *Google Safe Browsing*; by the time such URLs would get blacklisted, attackers would have shifted their websites

to alternate paths or subdomains on the same web server. Enhanced phishing reporting protocols—ideally bolstered by trust between vetted entities within the ecosystem—could help anti-phishing entities share detailed attack data at scale. Without sacrificing user privacy, sufficiently detailed information should be shared (e.g., request parameters, redirection chains, or a screenshot) [23]. This would be a key step toward increasing the effectiveness of blacklists against evasive phishing.

Similarly, the ecosystem currently lacks *standardized* approaches for malicious content take-down [1]. Although major hosting providers may have well-documented avenues for removing phishing websites, attackers flourish by using bulletproof hosting [65] or small hosts with fewer resources for timely intervention, or by compromising infrastructure.

**Phishing Links in E-mails:** Attackers make heavy use of redirection links in phishing e-mail lures. As I have shown, such links complicate the correlation of phishing e-mails with live websites—and, in turn, hamper further mitigation efforts, such as blacklisting. During my data collection period, I found that only 1.98% of links in lures were blacklisted by *Google Safe Browsing* at the time the report was received [4].

Additionally, I observed an average delay of 9.62 hours between the start of each phishing e-mail campaign (i.e., the initial arrival of a phishing message) and the *first* report sent by a victim. Due to this delay, I believe that direct user reports should only serve as a secondary means for entities to discover new phishing attacks. E-mail providers who directly receive phishing messages are in a better position to respond quickly, and could positively impact the ecosystem by directly sharing URLs with the organizations which they target, or with clearinghouses, when sufficient confidence exists [16].

In my dataset, at the granularity of individual phishing hostnames, e-mail lures were sent in large spikes, similar to what has been previously observed [76]. If a message is

[4] A larger percentage of *landing* URLs may have been blacklisted, however.

initially classified as benign but the URL within it is later detected as phishing, the e-mail should retroactively be hidden from the user by the e-mail system.

### *6.8.2 Third-party Resources*

It may seem counter-intuitive for malicious websites to embed external web resources hosted (and trackable) by third parties, especially in light of my findings that these resources enable both analysis and mitigation of phishing attacks. However, I argue that phishing websites will nevertheless use external resources for several reasons.

Most importantly, anti-phishing systems use known filenames of scripts, images, favicons, and archives as one type of fingerprinting to identify malicious websites [42, 102]. Phishing pages which only link to external files can avoid such fingerprinting entirely; with the added use of cloaking on their landing pages, phishing websites can remain stealthy to avoid or delay detection by the ecosystem.

In some cases, attackers choose to use third-party *services* for their own benefit. Within my dataset, the use of reCAPTCHA is one such example. Additionally, I observed phishing websites hosted on single-page *pastebin* services [77]. In order for phishing pages to render correctly on such services (and thus successfully fool victims), most images and scripts must be retrieved from external sources.

The use of external files can also ensure consistency between the look and feel of the legitimate website and a phishing page. Phishing kits can thus remain current without the need for frequent updates, which may be particularly desirable for phishers who do not want to invest money into sophisticated phishing kits. It is also easier for attackers to directly copy the source of the original page than to build a deceptive version from scratch.

Even if they do not embed third-party resources, phishing websites may link back to the legitimate organization's website and could thus be detected by my approach. The

same applies if victims are redirected back to the legitimate website after being phished: a common strategy used by attackers to minimize victims' awareness of the attack.

### *6.8.3 Ethical Considerations*

I took great care to ensure that user privacy was preserved throughout this research. My analysis did not involve access to any PII, and processing which entailed datasets that could contain PII (such as user account information or e-mail report content) was carried out in a purely programmatic fashion by existing, automated systems. During my analysis of user account compromise times, I only handled anonymized session or account identifiers which were interpreted and aggregated by the organization with which I collaborated.

Entities that become aware of compromised accounts within their systems—through internal or external data sources—should make reasonable efforts to re-secure such accounts [125]. During my research, I ensured that user accounts that I associated with phishing website traffic by *Golden Hour* were appropriately flagged by the organization. Furthermore, I recommended that the organization investigate accounts associated with users who likely visited a sophisticated phishing website, in an effort to identify and thwart the underlying threat actors.

## 6.9 Limitations

My analysis should be considered alongside certain limitations. Despite a large sample size, my data is based on victim traffic to phishing websites that target a single organization, which may skew my findings. However, the *Golden Hour* framework is not tied to any individual organization; thus, future analysis of other datasets could deepen the degree of insight into the ecosystem.

Due to the nature of my agreements with the organization, I cannot disclose certain concrete findings from our analysis, such as the total profit secured by attackers. Also, the success of phishing attacks hinges on numerous factors—such as the content and type of the original lure, appearance of the landing page, or redirection services used—which I did not consider, but which could provide details about the ecosystem vulnerabilities being exploited.

Despite the incentives for phishers to use third-party resources, as discussed in Section 6.8.2, my approach does not guarantee the detectability of an arbitrary phishing website. Phishers could deliberately evade my approach by excluding any trackable third-party files and avoiding redirecting victims back to the organization's website.

The time frame of my e-mail dataset is shorter than that of my web event dataset. I originally intended to correlate the event data with phishing URLs sent to victim inboxes at a major e-mail provider over the full data collection period. However, the prevalence of redirection links within phishing e-mails made such correlation difficult to scale, and I was not able to develop an alternative approach within the parameters of the established data-sharing agreement.

Lastly, my web event correlation approach (stage ❸ of the framework) benefits from the ability to accurately classify URLs as suspicious from within a large stream of traffic data, or a reliable source of ground truth (i.e., known phishing URLs) to match events. I only did the latter during my deployment; however, I consider my data sources (in Section 6.4.1) to be of high quality: peaks in *Crawler* traffic in my event dataset coincided with detection times of URLs in the phishing feeds considered. Yet, recent research has shown that even reputable anti-phishing vendors fail to identify many of the phishing URLs reported to them [106]. Future deployments of my approach should maximize the number of data sources for correlation to further increase *visibility*.

## 6.10 Related Work

Because phishing attacks are by nature spread across diverse infrastructure, empirical measurements of the relationships between the different attack phases are difficult. Nevertheless, such measurements can deliver crucial insights that are not possible at a finer granularity. To the best of my knowledge, this work is the first to paint an end-to-end picture of phishing attacks at scale by correlating victim traffic to live phishing websites with attack distribution and monetization.

The work most closely related to what I presented in this chapter is that of Heijden and Allodi [130], who leveraged methodology similar to *Golden Hour* to correlate URLs in phishing e-mails (reported to an organization) with the timestamps of clicks by individual victims. The authors combined the click data with e-mail content analysis to identify cognitive and technical factors that characterize successful phishing e-mails, which can help prioritize the mitigation of high-impact phishing URLs.

Han et al. monitored the life cycles of phishing kits installed on a honeypot [51]. Unlike my approach, the authors captured the credentials sent by each phishing kit and more closely analyzed attackers' interaction with the kit. However, honeypots are limited in scale and scope compared to my approach and do not offer insight into the damage caused by phishing, such as how stolen credentials are ultimately used.

Thomas et al. [125] analyzed a one-year dataset of data breaches, phished credentials, and keyloggers to study trends in the users victimized by such attacks, and the effectiveness of each type of attack. Although this work did not strictly focus on phishing attack anatomy, it underscores the effectiveness of large-scale, cross-organizational data analysis to capture the state of the ecosystem.

Ho et al. [55] analyzed over 113 million e-mails sent by employees of enterprise organizations to model lateral phishing attacks carried out via compromised e-mail accounts.

The authors revealed new types of attacks marked by both sophistication and effectiveness. Although this work does not focus on traditional phishing, it shows that important insight can be gained from analyzing attack data at scale.

Other prior work has scrutinized the time between phishing attack detection and blacklisting [98]. In Chapters 4 and 5, I conducted a controlled empirical analysis of the effectiveness of evasion techniques against the response time and coverage of browser anti-phishing blacklists [101]. The studies revealed weaknesses in blacklists and measured the gap between attack detection and mitigation under specific conditions: a gap which I also observed using the *Golden Hour* dataset.

In early measurements of the ecosystem, Moore and Clayton analyzed the temporal relationship between the sending of spam e-mails and the availability of phishing websites [92], as well as the latency between phishing deployment and detection by anti-phishing blacklists [91]. The authors cited a need for take-down due to the persistence of spam campaigns.

## 6.11 Summary

At their disposal, phishers have an array of sophisticated techniques that aim to circumvent existing anti-phishing defenses and increase the likelihood of compromising victims. With the addition of underground resources, such attacks are scalable, as has long been observed by the ecosystem [6]. However, the ecosystem itself is not powerless to fight back, as it has access to a wealth of data that can be used to analyze, detect, and prevent phishing. By correlating data from multiple ecosystem sources, I performed a longitudinal, end-to-end life cycle analysis of phishing attacks on a large scale: I not only gained insight into the timing of key events associated with modern phishing attacks, but also identified the gaps in defenses that phishers actively target. Furthermore, by analyzing the *Golden Hour* dataset discussed in this chapter, I was able to replicate the gen-

eral findings of the empirical measurements originally made by *PhishFarm* and *PhishTime* (Chapters 4 and 5), which further speaks to the importance of strengthening weaknesses within the ecosystem.

Phishing remains a significant threat to Internet users in part because the *reactive* anti-phishing defenses that are standard throughout the ecosystem, such as blacklists, struggle to effectively address the agility and sophistication of attackers. Importantly, analysis such as that carried out in my research can inform anti-phishing entities of an appropriate response time threshold for specific mitigations, to ultimately narrow the window of opportunity available to phishers.

My use of the *Golden Hour* framework to automatically secure the accounts of tens of thousands of phishing victims also motivates the continued expansion of *proactive* mitigations within the ecosystem. The framework could be practically adapted by any organization (commonly targeted by phishers) that tracks its own phishing URL and web traffic data, and can help seal gaps in defenses by securing compromised user accounts and enabling the earlier detection of phishing websites. Moreover, closer collaboration between anti-phishing entities, coupled with the development of enhanced and standardized mechanisms for sharing intelligence, would allow such mitigations to better scale to the ecosystem level.

# Chapter 7

# CONCLUSION

Through the work presented in this dissertation, I have examined phishing attacks from multiple angles, which, collectively, provide critical insight into ways in which the security community can continue improving its defenses to protect Internet users from phishing. Security recommendations made as a result of my work have already directly helped strengthen some of these defenses, such as browser blacklists on mobile devices[1], the data sources and detection strategies used by blacklists[2], and proactive anti-phishing mitigations[3].

I began my analysis from the perspective of phishers' own tools, which revealed the widespread and commoditized use of evasion techniques in phishing websites. I then developed and executed methodology (*PhishFarm*) to measure how well browser blacklists (a key ecosystem defense)—and specific anti-phishing entities—detect and mitigate these evasive websites.

Next, I generalized my methodology into a framework (*PhishTime*) for *continuously* measuring the performance of the anti-phishing ecosystem, in the long term, by identifying and replicating the characteristics of sophisticated phishing attacks being launched by criminals. Through a one-year study, I found that although the ecosystem had addressed some of the vulnerabilities identified by PhishFarm, other exploitable gaps remained. These gaps help explain why phishers remain persistent in their attacks.

[1] See Section 4.6.1.

[2] See Section 5.11.

[3] See Section 6.7.2.

Finally, I developed a novel and practical approach (*Golden Hour*) for building a meaningful dataset of real victim traffic to live phishing websites. I leveraged this dataset to perform the first end-to-end analysis of modern phishing attacks: from the time of e-mail distribution, to the victim interaction with the phishing website, to account monetization. This analysis not only validated many of the empirical findings from Chapters 3-5, but also shed light on the real-world damage that evasive phishing attacks cause. Moreover, my approach inspired me to propose a proactive mitigation strategy that can help organizations secure the online accounts of users who have been tricked by phishers, such that those accounts cannot subsequently be monetized by criminals.

## 7.1 Remaining Challenges in Anti-phishing

The modern anti-phishing ecosystem represents a considerable improvement over early decentralized mitigations [137], and it has continued to evolve—with notable changes even occurring as I conducted my research (between 2017 and 2019). The tremendous efforts made to date by security researchers and by industry should not be overlooked. However, I believe that many of the factors that have historically motivated phishers to carry out their attacks remain unchanged. Therefore, multi-layer mitigation strategies continue to be essential in the fight against phishing.

A key challenge within the ecosystem is the ease at which phishers can obtain infrastructure to carry out their attacks, whether it be through compromising legitimate web servers or partnering with service providers with lax abuse policies [1]. In some cases, the rise of legislation which promotes online privacy has also started hampering anti-abuse efforts, and may even help attackers cover their tracks [132]. At the same time, the distributed nature of phishing and the decentralized nature of reporting protocols makes it difficult to consistently track attackers from a law enforcement perspective.

As I have discussed throughout this dissertation, anti-phishing systems that operate at the ecosystem level are complex. Methods for evaluating these systems (e.g., PhishTime) are not currently standardized or used routinely and it has, therefore, been difficult to maintain an understanding of the effectiveness of current defenses and designate entities that could be held accountable for specific deficiencies. This situation is exacerbated by the difficulty of *measuring* the impact of phishing across the ecosystem as a whole: for example, organizations that know that their customers' credentials have been phished may seek to protect the accounts of these customers, but may not be able to prevent identity theft at other organizations even if they otherwise desire to do so (e.g., due to the lack of standard data-sharing protocols).

Because phishing is grounded in social engineering, training users to be aware of such attacks is an effective way to mitigate them [30, 68]. Yet, the awareness of the average Internet user remains insufficient [55, 103, 130] and is, thus, something that can also be promoted and improved upon across the ecosystem. More aware users would be more likely to spot and report phishing, which could speed up the ecosystem's *discovery* of attacks and lead to standardized approaches for sharing such reports between organizations to respond more comprehensively and track criminals more effectively.

## 7.2 Protecting the Internet of Tomorrow

It is my hope that beyond helping improve existing ecosystem defenses, the work presented in this dissertation inspires the next generation of anti-phishing systems as well as mitigations against broader online scams and other malicious content. For as long as phishing remains a problem, I believe that monitoring the ecosystem—similar to how PhishFarm and PhishTime operate—is critical to ensuring that adequate defenses are maintained. Although it may be difficult to ever fully prevent users from being deceived by social engineering tactics, technical systems should seek to tighten the noose around

the economics (i.e., profitability) of scams such as phishing. The cat-and-mouse game between attackers and defenders will continue indefinitely unless the ecosystem *treats* the underlying motivation of phishers, rather than the "symptoms" of phishing.

As we've observed with other types of cyber attacks [115], if and when phishing ceases being sufficiently profitable, attackers will likely abandon it and gravitate toward more viable alternatives. Within the realm of social engineering, attackers could turn to other types of scams that lack the same degree of scrutiny or ecosystem protection as phishing [50]. For example, even though today's browser-based anti-phishing systems block malicious (phishing) websites that impersonate well-known organizations, browsers do not block other fraudulent websites such as fake stores, high-yield investment schemes, or misleading services [8]. Likewise, scams that leverage communication channels other than e-mail, such as SMS text messages or phone calls, are considerably harder to track using current methods [128]. The ecosystem should, therefore, be prepared to adapt its defenses to new social engineering attack variants.

To reduce the aforementioned profitability of phishing, the ecosystem can work to systematically hamper the scalability of phishing attacks (from the perspective of web infrastructure) while simultaneously reducing the intrinsic value of stolen information. For example, initiatives such as passwordless authentication [63] can eliminate the need for—and many of the problems associated with—usernames and passwords [125]. A similar approach could be taken with payment information like credit card numbers. Moreover, new methods for enhanced validation of user identities and other sensitive data, in both online and offline contexts, could help mitigate identity theft and nullify unsolicited information requests from attackers. Universal adoption of these next-generation authentication and validation schemes is far from straightforward, but may be essential to the security of computer systems in the long term, and thus represents a key future research direction.

[1] Alowaisheq, E., P. Wang, S. Alrwais, X. Liao, X. Wang, T. Alowaisheq, X. Mi, S. Tang and B. Liu, "Cracking the wall of confinement: Understanding and analyzing malicious domain take-downs", in "Proceedings of the Network and Distributed System Security Symposium (NDSS)", (2019).

[2] Anderson, B., "Best browser automation testing tools for 2018 (top 10 reviews)", URL https://medium.com/@briananderson2209/best-automation-testing-tools-for-2018-top-10-reviews-8a4a19f664d2 (2017).

[3] APWG, "Anti-Phishing Working Group: APWG Phishing Activity trends Report- October 2004", URL https://docs.apwg.org/reports/APWG_Phishing_Activity_Report-Oct2004.pdf, (Date last accessed 25-August-2019) (2004).

[4] APWG, "Anti-Phishing Working Group: APWG Trends Report Q4 2016", URL https://docs.apwg.org/reports/apwg_trends_report_q4_2016.pdf, (Date last accessed 23-August-2017) (2016).

[5] APWG, "Anti-Phishing Working Group: APWG Trends Report Q1 2018", URL https://docs.apwg.org/reports/apwg_trends_report_q1_2018.pdf, (Date last accessed 31-August-2018) (2018).

[6] APWG, "Anti-Phishing Working Group: APWG Trends Report Q1 2019", URL https://docs.apwg.org/reports/apwg_trends_report_q1_2019.pdf, (Date last accessed 21-August-2019) (2019).

[7] APWG, "Anti-Phishing Working Group: APWG Trends Report Q3 2019", URL https://docs.apwg.org/reports/apwg_trends_report_q3_2019.pdf, (Date last accessed 6-November-2019) (2019).

[8] Beals, M., M. DeLiema and M. Deevy, "Framework for a taxonomy of fraud", Washington DC: Stanford Longevity Center/FINRA Financial Investor Education Foundation/Fraud Research Center. Retrieved August **25**, 2016 (2015).

[9] Berghel, H., "Equifax and the latest round of identity theft roulette", Computer **50**, 12, 72–76 (2017).

[10] Berners-Lee, T., L. Masinter and M. McCahill, "Uniform resource locators (URL)", Tech. rep. (1994).

[11] Bilge, L., E. Kirda, C. Kruegel and M. Balduzzi, "Exposure: Finding malicious domains using passive dns analysis.", in "Proceedings of the Network and Distributed System Security Symposium (NDSS)", (2011).

[12] Birk, D., S. Gajek, F. Grobert and A. R. Sadeghi, "Phishing phishers - observing and tracing organized cybercrime", in "Second International Conference on Internet Monitoring and Protection (ICIMP 2007)", p. 3 (2007).

[13] Blum, A., B. Wardman, T. Solorio and G. Warner, "Lexical feature based phishing url detection using online learning", in "Proceedings of the 3rd ACM Workshop on Artificial Intelligence and Security", AISec '10, pp. 54–60 (ACM, New York, NY, USA, 2010).

[14] Borgolte, K., C. Kruegel and G. Vigna, "Meerkat: Detecting website defacements through image-based object recognition", in "Proceedings of the 24th USENIX Security Symposium", pp. 595–610 (2015).

[15] Brinskelle, J. E., "Enforcement of same origin policy for sensitive data", US Patent 8,856,869 (2014).

[16] Butler, B., B. Wardman and N. Pratt, "Reaper: an automated, scalable solution for mass credential harvesting and osint", in "2016 APWG symposium on electronic crime research (eCrime)", pp. 1–10 (IEEE, 2016).

[17] Canali, D., D. Balzarotti and A. Francillon, "The role of web hosting providers in detecting compromised websites", in "Proceedings of the 22nd International Conference on World Wide Web", WWW '13, pp. 177–188 (ACM, New York, NY, USA, 2013).

[18] Chen, T.-C., T. Stepan, S. Dick and J. Miller, "An anti-phishing system employing diffused information", ACM Transactions on Information and System Security (TISSEC) **16**, 4, 16 (2014).

[19] Chhabra, S., A. Aggarwal, F. Benevenuto and P. Kumaraguru, "Phi.sh/$ocial: the phishing landscape through short URLs", in "Proceedings of the 8th Annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference", pp. 92–101 (ACM, 2011).

[20] Cohen, J., "Statistical power analysis for the behavioral sciences. 2nd", (1988).

[21] Costello, A., "Punycode: A bootstring encoding of unicode for internationalized domain names in applications (IDNA)", RFC 3492 (2003).

[22] Cova, M., C. Kruegel and G. Vigna, "There is no free phish: An analysis of "free" and live phishing kits", in "Proceedings of the 2nd Conference on USENIX Workshop on Offensive Technologies", WOOT, pp. 4:1–4:8 (Berkeley, CA, USA, 2008).

[23] CSSR, "Chrome suspicious site reporter", URL https://chrome.google.com/webstore/detail/suspicious-site-reporter/jknemblkbdhdcpllfgbfekkdciegfboi?hl=en-US (2019).

[24] Cui, Q., G.-V. Jourdan, G. V. Bochmann, R. Couturier and I.-V. Onut, "Tracking phishing attacks over time", in "Proceedings of the 26th International Conference on World Wide Web", pp. 667–676 (2017).

[25] CWE, "CWE-601: URL Redirection to Untrusted Site ('Open Redirect')", URL http://cwe.mitre.org/data/definitions/601.html (2018).

[26] Dalgic, F. C., A. S. Bozkir and M. Aydos, "Phish-iris: A new approach for vision based brand prediction of phishing web pages via compact visual descriptors", in "Proceedings of the 2nd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)", pp. 1–8 (IEEE, 2018).

[27] Dhamija, R., J. D. Tygar and M. Hearst, "Why phishing works", in "Proceedings of the SIGCHI Conference on Human Factors in Computing Systems", CHI, pp. 581–590 (ACM, New York, NY, USA, 2006).

[28] DigitalOcean, "Digitalocean: Cloud computing, simplicity at scale", URL https://www.digitalocean.com/ (2018).

[29] Dobolyi, D. G. and A. Abbasi, "Phishmonger: A free and open source public archive of real-world phishing websites", in "Intelligence and Security Informatics, 2016 IEEE Conference on", pp. 31–36 (IEEE, 2016).

[30] Dodge Jr, R. C., C. Carver and A. J. Ferguson, "Phishing for user security awareness", computers & security **26**, 1, 73–80 (2007).

[31] Drury, V. and U. Meyer, "Certified phishing: taking a look at public key certificates of phishing websites", in "15th Symposium on Usable Privacy and Security (SOUPSfi19). USENIX Association, Berkeley, CA, USA", pp. 211–223 (2019).

[32] Duman, S., K. Kalkan-Cakmakci, M. Egele, W. Robertson and E. Kirda, "Emailprofiler: Spearphishing filtering with header and stylometric features of emails", in "Proceedings of the Computer Software and Applications Conference (COMPSAC), 2016 IEEE 40th Annual", vol. 1, pp. 408–416 (IEEE, 2016).

[33] Emigh, A., "ITTC report on online identity theft technology and countermeasures 1: Phishing technology, chokepoints and countermeasures", Radix Labs (2005).

[34] ESET, "Eset: Report a phishing page", URL http://phishing.eset.com/report/ (2018).

[35] Esparza, J. M., "Understanding the credential theft lifecycle", Computer Fraud & Security **2**, 6–9 (2019).

[36] Evilginx, "Evilginx - Advanced Phishing with Two-factor Authentication Bypass", URL https://breakdev.org/evilginx-advanced-phishing-with-two-factor-authentication-bypass/ (2017).

[37] Fafinski, S., W. H. Dutton and H. Z. Margetts, "Mapping and measuring cybercrime", (2010).

[38] Fang, L., W. Bailing, H. Junheng, S. Yushan and W. Yuliang, "A proactive discovery and filtering solution on phishing websites", in "Big Data (Big Data), 2015 IEEE International Conference on", pp. 2348–2355 (IEEE, 2015).

[39] FBI, "Business e-mail compromise: The 12 billion dollar scam", https://www.ic3.gov/media/2018/180712.aspx , [Online; accessed 26-Aug-2019] (2018).

[40] Fielding, R., J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach and T. Berners-Lee, "Hypertext transfer protocol – HTTP/1.1", RFC 2616 (1999).

[41] Garera, S., N. Provos, M. Chew and A. D. Rubin, "A framework for detection and measurement of phishing attacks", in "Proceedings of the 2007 ACM Workshop on Recurring Malcode", WORM, pp. 1–8 (ACM, New York, NY, USA, 2007).

[42] Geng, G.-G., X.-D. Lee, W. Wang and S.-S. Tseng, "Favicon - a clue to phishing sites detection", in "Proceedings of the 2013 APWG Symposium on Electronic Crime Research (eCrime)", pp. 1–10 (IEEE, 2013).

[43] Google, "Google docs phishing campaign", URL https://www.us-cert.gov/ncas/current-activity/2017/05/04/Google-Docs-Phishing-Campaign (2017).

[44] Google, "Google safe browsing: Report phishing page", URL https://safebrowsing.google.com/safebrowsing/report_phish/ (2018).

[45] Google, "Better password protections in chrome", URL https://blog.google/products/chrome/better-password-protections/ (2019).

[46] Google, "Google safe browsing submission api", https://cloud.google.com/phishing-protection/docs/quickstart-submission-api, [Online; accessed 12-Dec-2019] (2019).

[47] Google, "Google safe browsing transparency report", URL https://transparencyreport.google.com/safe-browsing/overview?hl=en (2019).

[48] Google, "Overview — safe browsing apis (v4) — google developers", URL https://developers.google.com/safe-browsing/v4/ (2019).

[49] Gupta, B. B., N. A. Arachchilage and K. E. Psannis, "Defending against phishing attacks: taxonomy of methods, current issues and future directions", Telecommunication Systems **67**, 2, 247–267 (2018).

[50] Gupta, S., A. Singhal and A. Kapoor, "A literature survey on social engineering attacks: Phishing attack", in "2016 international conference on computing, communication and automation (ICCCA)", pp. 537–540 (IEEE, 2016).

[51] Han, X., N. Kheir and D. Balzarotti, "Phisheye: Live monitoring of sandboxed phishing kits", in "Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security", pp. 1402–1413 (ACM, 2016).

[52] Han, Y. and Y. Shen, "Accurate spear phishing campaign attribution and early detection", in "Proceedings of the 31st Annual ACM Symposium on Applied Computing", SAC '16, pp. 2079–2086 (ACM, New York, NY, USA, 2016).

[53] Hao, S., M. Thomas, V. Paxson, N. Feamster, C. Kreibich, C. Grier and S. Hollenbeck, "Understanding the domain registration behavior of spammers", in "Proceedings of the 2013 conference on Internet measurement", pp. 63–76 (ACM, 2013).

[54] Hilbert, M. and P. López, "The world's technological capacity to store, communicate, and compute information", science **332**, 6025, 60–65 (2011).

[55] Ho, G., A. Cidon, L. Gavish, M. Schweighauser, V. Paxson, S. Savage, G. M. Voelker and D. Wagner, "Detecting and characterizing lateral phishing at scale", in "Proceedings of the 28th USENIX Security Symposium", pp. 1273–1290 (2019).

[56] Holz, T., M. Engelberth and F. Freiling, "Learning more about the underground economy: A case-study of keyloggers and dropzones", Computer Security–ESORICS 2009 pp. 1–18 (2009).

[57] Hong, J., "The state of phishing attacks", Communications of the ACM **55**, 1, 74–81 (2012).

[58] Invernizzi, L., K. Thomas, A. Kapravelos, O. Comanescu, J.-M. Picod and E. Bursztein, "Cloak of visibility: Detecting when machines browse a different web", in "Proceedings of the 37th IEEE Symposium on Security and Privacy", (2016).

[59] IP2Location, "Ip2location.com", URL https://www.ip2location.com/, (Date last accessed 30-Nov-2018) (2018).

[60] ISC, "Internet Domain Survey, January, 2019", URL https://downloads.isc.org/www/survey/reports/current/ (2019).

[61] Jackson, C., D. R. Simon, D. S. Tan and A. Barth, "An evaluation of extended validation and picture-in-picture phishing attacks", in "International Conference on Financial Cryptography and Data Security", pp. 281–293 (Springer, 2007).

[62] Kaspersky, "Spam and phishing in Q1 2019", URL https://securelist.com/spam-and-phishing-in-q1-2019/90795/, (Date last accessed 25-August-2019) (2019).

[63] Khalil, M. M. and V. R. Challa, "Password-less authentication service", US Patent 9,537,661 (2017).

[64] Khonji, M., Y. Iraqi and A. Jones, "Enhancing phishing e-mail classifiers: A lexical url analysis approach", International Journal for Information Security Research (IJISR) **2**, 1/2, 40 (2012).

[65] Konte, M., R. Perdisci and N. Feamster, "Aswatch: An as reputation system to expose bulletproof hosting ases", ACM SIGCOMM Computer Communication Review **45**, 4, 625–638 (2015).

[66] Krombholz, K., H. Hobel, M. Huber and E. Weippl, "Advanced social engineering attacks", Journal of Information Security and applications **22**, 113–122 (2015).

[67] Kshetri, N., *Cybercrime and cybersecurity in the global south* (Springer, 2013).

[68] Kumaraguru, P., J. Cranshaw, A. Acquisti, L. Cranor, J. Hong, M. A. Blair and T. Pham, "School of phish: a real-world evaluation of anti-phishing training", in "Proceedings of the 5th Symposium on Usable Privacy and Security", pp. 1–12 (2009).

[69] Let's Encrypt, "Let's Encrypt No Longer Checking Google Safe Browsing", URL https://community.letsencrypt.org/t/let-s-encrypt-no-longer-checking-google-safe-browsing/82168, (Date last accessed 10-January-2019) (2019).

[70] Lewis, E., "The role of wildcards in the domain name system", RFC 4592 (2006).

[71] Liang, B., M. Su, W. You, W. Shi and G. Yang, "Cracking classifiers for evasion: A case study on Google's phishing pages filter", in "Proceedings of the 25th International Conference on World Wide Web", pp. 345–356 (2016).

[72] Ludl, C., S. McAllister, E. Kirda, e. H. B. Kruegel, Christopher and R. Sommer, "On the effectiveness of techniques to detect phishing sites", in "Detection of Intrusions and Malware, and Vulnerability Assessment", pp. 20–39 (Springer Berlin Heidelberg, 2007).

[73] Luo, M., O. Starov, N. Honarmand and N. Nikiforakis, "Hindsight: Understanding the evolution of UI vulnerabilities in mobile browsers", in "Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security", pp. 149–162 (ACM, 2017).

[74] Manky, D., "Cybercrime as a service: a very modern business", Computer Fraud & Security **2013**, 6, 9–13 (2013).

[75] Marchal, S. and N. Asokan, "On designing and evaluating phishing webpage detection techniques for the real world", in "Proceedings of the 11th USENIX Workshop on Cyber Security Experimentation and Test (CSET 18)", (USENIX Association, Baltimore, MD, 2018).

[76] Marchal, S., J. François, R. State and T. Engel, "Phishstorm: Detecting phishing with streaming analytics", IEEE Transactions on Network and Service Management **11**, 4, 458–471 (2014).

[77] Matic, S., A. Fattori, D. Bruschi and L. Cavallaro, "Peering into the muddy waters of pastebin", ERCIM News: Special Theme Cybercrime and Privacy Issues p. 16 (2012).

[78] Matsuoka, M., N. Yamai, K. Okayama, K. Kawano, M. Nakamura and M. Minda, "Domain registration date retrieval system of urls in e-mail messages for improving spam discrimination", in "Computer Software and Applications Conference Workshops (COMPSACW), 2013 IEEE 37th Annual", pp. 587–592 (IEEE, 2013).

[79] MaxMind, "Geolite2 database", URL https://dev.maxmind.com/geoip/geoip2/geolite2/, (Date last accessed 30-Nov-2018) (2018).

[80] McAfee, "Mcafee: Custom url ticketing system", URL https://www.trustedsource.org/en/feedback/url?action=checksingle (2018).

[81] McAfee, "Economic Impact of Cybercrime- No Slowing Down", https://www.mcafee.com/enterprise/en-us/assets/reports/restricted/rp-economic-impact-cybercrime.pdf, [Date last accessed 25-August-2019] (2019).

[82] McCalley, H., B. Wardman and G. Warner, "Analysis of back-doored phishing kits.", in "IFIP Int. Conf. Digital Forensics", vol. 361, pp. 155–168 (Springer, 2011).

[83] Metcalfe, R., "The stockings were hung by the chimney with care", RFC 602 (1973).

[84] Microsoft, "Smartscreen: Report a website", URL https://feedback.smartscreen.microsoft.com/feedback.aspx?t=0&url= (2018).

[85] Miniwatts, "World Internet Users and 2018 Population Stats", URL https://www.internetworldstats.com/stats.htm (2018).

[86] Miramirkhani, N., T. Barron, M. Ferdman and N. Nikiforakis, "Panning for gold.com: Understanding the dynamics of domain dropcatching", in "Proceedings of the 2018 World Wide Web Conference", pp. 257–266 (2018).

[87] Mockapetris, P., "Domain names - implementation and specification", RFC 1034 (1987).

[88] Mohurle, S. and M. Patil, "A brief study of Wannacry threat: Ransomware attack 2017", International Journal of Advanced Research in Computer Science **8**, 5 (2017).

[89] Moore, T. and R. Clayton, "Examining the impact of website take-down on phishing", in "Proceedings of 2007 APWG Symposium on Electronic Crime Research (eCrime)", pp. 1–13 (ACM, 2007).

[90] Moore, T. and R. Clayton, "How hard can it be to measure phishing?", Mapping and Measuring Cybercrime (2010).

[91] Moore, T. and R. Clayton, "Discovering phishing dropboxes using email metadata", in "Proceedings of the 2012 APWG Symposium on Electronic Crime Research (eCrime)", pp. 1–9 (IEEE, 2012).

[92] Moore, T., R. Clayton and H. Stern, "Temporal correlations between spam and phishing websites.", in "LEET", (2009).

[93] Myers, M., R. Ankney, A. Malpani, S. Galperin and C. Adams, "X. 509 internet public key infrastructure online certificate status protocol-ocsp", Tech. rep., RFC 2560 (1999).

[94] Nero, P. J., B. Wardman, H. Copes and G. Warner, "Phishing: Crime that pays", in "2011 APWG Symposium on Electronic Crime Research (eCrime)", pp. 1–10 (IEEE, 2011).

[95] NetApplications, "Browser market share", https://netmarketshare.com/browser-market-share.aspx, [Online; accessed 1-Feb-2019] (2019).

[96] NetCraft, "NetCraft August 2017 Web Server Survey", URL https://news.netcraft.com/archives/2017/08/29/august-2017-web-server-survey.html (2017).

[97] NetCraft, "NetCraft: Report a Phishing URL", URL http://toolbar.netcraft.com/report_url (2018).

[98] NSS Labs, "Web browser security: Phishing protection test methodology v3.0", URL https://research.nsslabs.com/reports/free-90/files/TestMethodology_WebB/Page4 (2016).

[99] NSS Labs, "Nss labs conducts first cross-platform test of leading web browsers", URL https://www.nsslabs.com/company/news/press-releases/nss-labs-conducts-first-cross-platform-test-of-leading-web-browsers/ (2017).

[100] Nykvist, C., L. Sjöström, J. Gustafsson and N. Carlsson, "Server-side adoption of certificate transparency", in "Proceedings of the International Conference on Passive and Active Network Measurement", pp. 186–199 (Springer, 2018).

[101] Oest, A., Y. Safaei, A. Doupé, G. Ahn, B. Wardman and K. Tyers, "Phishfarm: A scalable framework for measuring the effectiveness of evasion techniques against browser phishing blacklists", in "Proceedings of the 2019 IEEE Symposium on Security and Privacy", pp. 764–781 (2019).

[102] Oest, A., Y. Safaei, A. Doupé, G. Ahn, B. Wardman and G. Warner, "Inside a phisher's mind: Understanding the anti-phishing ecosystem through phishing kit analysis", in "Proceedings of the 2018 APWG Symposium on Electronic Crime Research (eCrime)", pp. 1–12 (2018).

[103] Oest, A., P. Zhang, B. Wardman, E. Nunes, J. Burgis, A. Zand, K. Thomas, A. Doupé and G.-J. Ahn, "Sunrise to sunset: Analyzing the end-to-end life cycle and effectiveness of phishing attacks at scale", in "Proceedings of the 29th USENIX Security Symposium", (2020).

[104] OpenPhish, "OpenPhish", URL https://openphish.com (2019).

[105] PayPal, "Report a suspicious e-mail or website.", https://www.paypal.com/us/webapps/mpp/security/report-problem, [Online; accessed 20-Feb-2019] (2019).

[106] Peng, P., L. Yang, L. Song and G. Wang, "Opening the blackbox of virustotal: Analyzing online phishing scan engines", in "Proceedings of the 2019 conference on Internet measurement (IMC)", (ACM, 2019).

[107] PhishTank, "PhishTank", URL https://phishtank.com (2018).

[108] Postel, J., "Internet protocol", RFC 791 (1981).

[109] Provos, N., J. McClain and K. Wang, "Search worms", in "Proceedings of the 4th ACM Workshop on Recurring Malcode", WORM '06, pp. 1–8 (ACM, New York, NY, USA, 2006).

[110] Ramzan, Z., "Phishing attacks and countermeasures", in "Handbook of information and communication security", pp. 433–448 (Springer, 2010).

[111] Randy Abrams, J. P., Orlando Barrera, "Browser Security Comparative Analysis - Phishing Protection", NSS Labs URL https://www.helpnetsecurity.com/images/articles/BrowserSecurityCAR2013-PhishingProtection.pdf (2013).

[112] Ravishankar, K., B. Prasad, S. Gupta and K. K. Biswas, "Dominant color region based indexing for cbir", in "Image Analysis and Processing, 1999. Proceedings. International Conference on", pp. 887–892 (IEEE, 1999).

[113] Ruxton, G. D., "The unequal variance t-test is an underused alternative to student's t-test and the mann–whitney u test", Behavioral Ecology **17**, 4, 688–690 (2006).

[114] Sheng, S., B. Wardman, G. Warner, L. F. Cranor, J. Hong and C. Zhang, "An empirical analysis of phishing blacklists", in "Proceedings of the Sixth Conference on Email and Anti-Spam (CEAS)", (2009).

[115] Shoshitaishvili, Y., R. Wang, C. Salls, N. Stephens, M. Polino, A. Dutcher, J. Grosen, S. Feng, C. Hauser, C. Kruegel *et al.*, "Sok:(state of) the art of war: Offensive techniques in binary analysis", in "2016 IEEE Symposium on Security and Privacy (SP)", pp. 138–157 (IEEE, 2016).

[116] Siadati, H., S. Palka, A. Siegel and D. McCoy, "Measuring the effectiveness of embedded phishing exercises", in "10th USENIX Workshop on Cyber Security Experimentation and Test (CSET 17)", (2017).

[117] SMTP, "Simple Mail Transfer Protocol", RFC 821 (1982).

[118] Sood, A. K. and R. J. Enbody, "Crimeware-as-a-service: a survey of commoditized crimeware in the underground market", International Journal of Critical Infrastructure Protection **6**, 1, 28–38 (2013).

[119] StatCounter, "Statcounter: Desktop browser market share worldwide", http://gs.statcounter.com/browser-market-share/, [Online; accessed 20-Aug-2017] (2017).

[120] StatCounter, "Desktop vs mobile vs tablet market share worldwide", http://gs.statcounter.com/platform-market-share/desktop-mobile-tablet, [Online; accessed 01-Nov-2019] (2019).

[121] Stringhini, G., C. Kruegel and G. Vigna, "Shady paths: Leveraging surfing crowds to detect malicious web pages", in "Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security", CCS '13, pp. 133–144 (ACM, New York, NY, USA, 2013).

[122] Szurdi, J., B. Kocso, G. Cseh, J. Spring, M. Felegyhazi and C. Kanich, "The long fitailefi of typosquatting domain names", in "23rd USENIX Security Symposium", pp. 191–206 (2014).

[123] Takata, Y., S. Goto and T. Mori, "Analysis of redirection caused by web-based malware", Proceedings of the Asia-Pacific advanced network **32**, 53–62 (2011).

[124] Tencent, "Tencent's 'two-pronged approach' joins apple to solve information harassment problem", https://www.qq.com/a/20170614/059855.htm, [Online; accessed 20-Feb-2019] (2017).

[125] Thomas, K., F. Li, A. Zand, J. Barrett, J. Ranieri, L. Invernizzi, Y. Markov, O. Comanescu, V. Eranti, A. Moscicki *et al.*, "Data breaches, phishing, or malware?: Understanding the risks of stolen credentials", in "Proceedings of the 2017 ACM SIGSAC conference on computer and communications security", pp. 1421–1434 (ACM, 2017).

[126] Thomas, K., D. McCoy, C. Grier, A. Kolcz and V. Paxson, "Trafficking fraudulent accounts: The role of the underground market in Twitter spam and abuse.", in "Proceedings of the 22nd USENIX Security Symposium", pp. 195–210 (2013).

[127] Tian, K., S. T. Jan, H. Hu, D. Yao and G. Wang, "Needle in a haystack: tracking down elite phishing domains in the wild", in "Proceedings of the Internet Measurement Conference 2018", pp. 429–442 (ACM, 2018).

[128] Tu, H., A. Doupé, Z. Zhao and G.-J. Ahn, "Users really do answer telephone scams", in "Proceedings of the 28th USENIX Security Symposium", pp. 1327–1340 (2019).

[129] Ulqinaku, E., D. Lain and S. Capkun, "2fa-pp: 2nd factor phishing prevention", in "Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks", pp. 60–70 (ACM, 2019).

[130] Van der Heijden, A. and L. Allodi, "Cognitive triaging of phishing attacks", in "Proceedings of the 28th USENIX Security Symposium", (2019).

[131] Virvilis, N., N. Tsalis, A. Mylonas and D. Gritzalis, "Mobile devices: A phisher's paradise", 2014 11th International Conference on Security and Cryptography (SECRYPT) pp. 1–9 (2014).

[132] Voigt, P. and A. Von dem Bussche, "The eu general data protection regulation (gdpr)", A Practical Guide, 1st Ed., Cham: Springer International Publishing (2017).

[133] Von Ahn, L., B. Maurer, C. McMillen, D. Abraham and M. Blum, "recaptcha: Human-based character recognition via web security measures", Science **321**, 5895, 1465–1468 (2008).

[134] Wardman, B., "Assessing the gap: Measure the impact of phishing on an organization", Annual Conference on Digital Forensics, Security and Law (2016).

[135] Weimer, F., "Passive dns replication", in "FIRST conference on computer security incident", p. 98 (2005).

[136] Whittaker, C., B. Ryner and M. Nazif, "Large-scale automatic classification of phishing pages", in "Proceedings of the Network and Distributed System Security Symposium (NDSS)", (2010).

[137] Wu, M., R. C. Miller and S. L. Garfinkel, "Do security toolbars actually prevent phishing attacks?", in "Proceedings of the SIGCHI Conference on Human Factors in Computing Systems", CHI '06, pp. 601–610 (ACM, New York, NY, USA, 2006).

[138] Xiang, G., J. Hong, C. P. Rose and L. Cranor, "Cantina+: A feature-rich machine learning framework for detecting phishing web sites", ACM Trans. Inf. Syst. Secur. (2011).

[139] Xu, W., F. Zhang and S. Zhu, "The power of obfuscation techniques in malicious javascript code: A measurement study", in "2012 7th International Conference on Malicious and Unwanted Software", pp. 9–16 (IEEE, 2012).

[140] Yandex, "Safe browsing api", https://tech.yandex.com/safebrowsing/, [Online; accessed 20-Feb-2019] (2019).

[141] Yang, W., A. Xiong, J. Chen, R. W. Proctor and N. Li, "Use of phishing training to improve security warning compliance: Evidence from a field experiment", in "Proceedings of the Hot Topics in Science of Security: Symposium and Bootcamp", HoTSoS, pp. 52–61 (ACM, New York, NY, USA, 2017).

[142] Zhang, Y., J. I. Hong and L. F. Cranor, "Cantina: A content-based approach to detecting phishing web sites", in "Proceedings of the 16th International Conference on World Wide Web", WWW, pp. 639–648 (ACM, New York, NY, USA, 2007).

# APPENDIX A

# PRELIMINARY PHISHFARM TEST DATA

Table A.1 includes detailed performance scores for all entities in the preliminary tests. These scores are based on the formulas in Section 4.5.2 and are the basis of the comparative discussion in Section 4.5.3. I visualize the preliminary test performance of only the subsequently re-tested entities (GSB, SmartScreen, AWPG, PhishTank, and PayPal) in Figure A.1a.

Figure A.1b illustrates the increased likelihood of URLs with a deceptive domain (Type IV [41, 102]) to be blacklisted during the preliminary tests. As previously discussed, this increase is linked to heuristics used by SmartScreen browsers; the positive effect that this had on IE and Edge blacklisting can be seen in the browser performance breakdown in Figure A.1c. The latter two charts are based on data from all 10 preliminary tests.

| **GSB** | *Filter A* | *Filter B* | *Filter C* | *Filter D* | *Filter E* | *Filter F* | $S_b$ | |
|---|---|---|---|---|---|---|---|---|
| $S_{bf}$ *GSB* | 0.988 | 0 | 0 | 0.846 | 0 | 0.493 | 0.466 | |
| $S_{bf}$ *IE* | 0 | 0.142 | 0 | 0 | 0 | 0.148 | 0.049 | |
| $S_{bf}$ *Edge* | 0.950 | 0.130 | 0 | 0.709 | 0 | 0.703 | 0.551 | |
| $S_{bf}$ *Opera* | 0.138 | 0 | 0 | 0.236 | 0 | 0 | 0.046 | |
| $PB_f$ | 1.000 | 1.000 | 0 | 0.857 | 0 | 0.833 | **0.421** | $S$ |
| $TB_f$ | 38 | 10 | N/A | 43 | N/A | 151 | 0.900 | $C$ |

| **SmartScreen** | *Filter A* | *Filter B* | *Filter C* | *Filter D* | *Filter E* | *Filter F* | $S_b$ | |
|---|---|---|---|---|---|---|---|---|
| $S_{bf}$ *GSB* | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| $S_{bf}$ *IE* | 0 | 0.142 | 0.284 | 0.142 | 0.135 | 0.166 | 0.100 | |
| $S_{bf}$ *Edge* | 0.956 | 0 | 0 | 0 | 0.952 | 0.703 | 0.870 | |
| $S_{bf}$ *Opera* | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| $PB_f$ | 1.000 | 0.143 | 0.286 | 0.143 | 1.000 | 0.833 | **0.045** | $S$ |
| $TB_f$ | 10 | 10 | 14 | 18 | 162 | 7 | 1 | $C$ |

| **APWG** | *Filter A* | *Filter B* | *Filter C* | *Filter D* | *Filter E* | *Filter F* | $S_b$ | |
|---|---|---|---|---|---|---|---|---|
| $S_{bf}$ *GSB* | 0.648 | 0 | 0 | 0.141 | 0 | 0 | 0.158 | |
| $S_{bf}$ *IE* | 0.255 | 0.432 | 0.306 | 0.306 | 0.524 | 0.178 | 0.319 | |
| $S_{bf}$ *Edge* | 0.958 | 0.142 | 0.137 | 0 | 0.821 | 0.632 | 0.804 | |
| $S_{bf}$ *Opera* | 0.345 | 0 | 0 | 0 | 0 | 0 | 0.115 | |
| $PB_f$ | 1.000 | 1.000 | 1.000 | 0.857 | 1.000 | 0.333 | **0.198** | $S$ |
| $TB_f$ | 73 | 286 | 199 | 154 | 276 | 188 | 1 | $C$ |

| **PhishTank** | *Filter A* | *Filter B* | *Filter C* | *Filter D* | *Filter E* | *Filter F* | $S_b$ | |
|---|---|---|---|---|---|---|---|---|
| $S_{bf}$ *GSB* | 0.971 | 0.141 | 0.261 | 0 | 0.399 | 0.303 | 0.387 | |
| $S_{bf}$ *IE* | 0.314 | 0 | 0 | 0 | 0.286 | 0.167 | 0.255 | |
| $S_{bf}$ *Edge* | 0.771 | 0 | 0 | 0.124 | 0.522 | 0.295 | 0.529 | |
| $S_{bf}$ *Opera* | 0.112 | 0.123 | 0 | 0 | 0 | 0 | 0.037 | |
| $PB_f$ | 1.000 | 0.286 | 0.286 | 0.125 | 0.714 | 0.333 | **0.372** | $S$ |
| $TB_f$ | 95 | 309 | 344 | 3784 | 162 | 363 | 0.975 | $C$ |

| **PayPal** | *Filter A* | *Filter B* | *Filter C* | *Filter D* | *Filter E* | *Filter F* | $S_b$ | |
|---|---|---|---|---|---|---|---|---|
| $S_{bf}$ *GSB* | 0.637 | 0 | 0.276 | 0 | 0.408 | 0 | 0.264 | |
| $S_{bf}$ *IE* | 0.345 | 0.517 | 0.448 | 0.306 | 0.525 | 0 | 0.290 | |
| $S_{bf}$ *Edge* | 0.213 | 0 | 0 | 0.173 | 0 | 0.145 | 0.119 | |
| $S_{bf}$ *Opera* | 0.102 | 0 | 0.253 | 0 | 0 | 0 | 0.034 | |
| $PB_f$ | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 0.167 | **0.255** | $S$ |
| $TB_f$ | 121 | 181 | 128 | 179 | 154 | 3694 | 0.925 | $C$ |

| **ESET** | *Filter A* | *Filter B* | *Filter C* | *Filter D* | *Filter E* | *Filter F* | $S_b$ | |
|---|---|---|---|---|---|---|---|---|
| $S_{bf}$ *GSB* | 0.297 | 0 | 0 | 0 | 0 | 0 | 0.059 | |
| $S_{bf}$ *IE* | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| $S_{bf}$ *Edge* | 0.256 | 0 | 0 | 0 | 0 | 0 | 0.085 | |
| $S_{bf}$ *Opera* | 0.137 | 0 | 0 | 0 | 0 | 0 | 0.046 | |
| $PB_f$ | 0.333 | 0 | 0 | 0 | 0 | 0 | **0.055** | $S$ |
| $TB_f$ | 444 | N/A | N/A | N/A | N/A | N/A | 1 | $C$ |

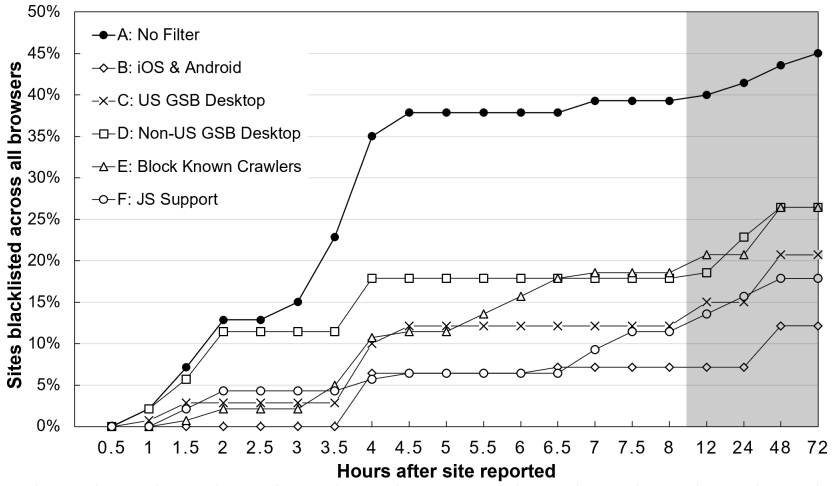
| **WebSense** | *Filter A* | *Filter B* | *Filter C* | *Filter D* | *Filter E* | *Filter F* | $S_b$ | |
|---|---|---|---|---|---|---|---|---|
| $S_{bf}$ *GSB* | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| $S_{bf}$ *IE* | 0 | 0 | 0.142 | 0 | 0 | 0.331 | 0.110 | |
| $S_{bf}$ *Edge* | 0 | 0 | 0.128 | 0 | 0 | 0.299 | 0.100 | |
| $S_{bf}$ *Opera* | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| $PB_f$ | 0 | 0 | 0.143 | 0 | 0 | 0.286 | **0.014** | $S$ |
| $TB_f$ | 444 | N/A | 4 | N/A | N/A | 6 | 0.275 | ***C*** |

| **Netcraft** | *Filter A* | *Filter B* | *Filter C* | *Filter D* | *Filter E* | *Filter F* | $S_b$ | |
|---|---|---|---|---|---|---|---|---|
| $S_{bf}$ *GSB* | 0 | 0.135 | 0 | 0.538 | 0 | 0 | 0.108 | |
| $S_{bf}$ *IE* | 0.166 | 0 | 0.142 | 0 | 0.134 | 0 | 0.100 | |
| $S_{bf}$ *Edge* | 0.389 | 0 | 0.129 | 0 | 0 | 0 | 0.130 | |
| $S_{bf}$ *Opera* | 0.302 | 0.260 | 0.130 | 0 | 0.130 | 0 | 0.144 | |
| $PB_f$ | 0.667 | 0.429 | 0.182 | 0.571 | 0.182 | 0 | **0.109** | $S$ |
| $TB_f$ | 531 | 334 | 206 | 241 | 318 | N/A | 0.975 | ***C*** |

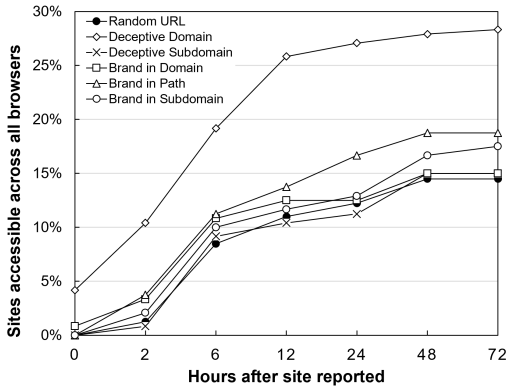
| **US CERT** | *Filter A* | *Filter B* | *Filter C* | *Filter D* | *Filter E* | *Filter F* | $S_b$ | |
|---|---|---|---|---|---|---|---|---|
| $S_{bf}$ *GSB* | 0 | 0 | 0.139 | 0 | 0 | 0 | 0.028 | |
| $S_{bf}$ *IE* | 0 | 0.127 | 0.142 | 0 | 0.134 | 0 | 0.045 | |
| $S_{bf}$ *Edge* | 0 | 0.127 | 0.127 | 0 | 0.127 | 0 | 0.042 | |
| $S_{bf}$ *Opera* | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| $PB_f$ | 0 | 0.143 | 0.286 | 0 | 0.143 | 0 | **0.029** | $S$ |
| $TB_f$ | N/A | 28 | 70 | N/A | 248 | N/A | 0.200 | ***C*** |

| **McAfee** | *Filter A* | *Filter B* | *Filter C* | *Filter D* | *Filter E* | *Filter F* | $S_b$ | |
|---|---|---|---|---|---|---|---|---|
| $S_{bf}$ *GSB* | 0 | 0 | 0 | 0 | 0 | 0.160 | 0.032 | |
| $S_{bf}$ *IE* | 0.167 | 0.127 | 0.143 | 0.143 | 0 | 0 | 0.056 | |
| $S_{bf}$ *Edge* | 0.939 | 0 | 0 | 0 | 0.955 | 0.118 | 0.671 | |
| $S_{bf}$ *Opera* | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| $PB_f$ | 1 | 0.143 | 0.143 | 0.143 | 1 | 0.167 | **0.059** | $S$ |
| $TB_f$ | 134 | 466 | 3702 | 3702 | 180 | 172 | 1 | ***C*** |

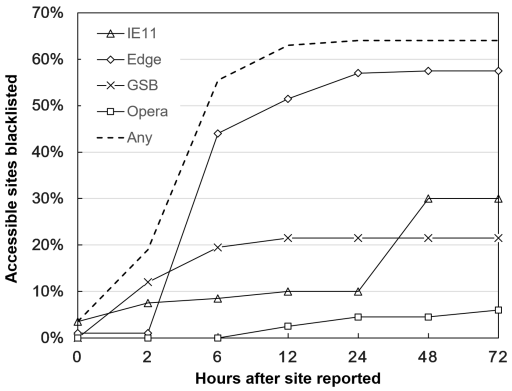
Table A.1: Aggregate entity blacklisting performance scores in the preliminary tests.




(a) By filter (re-tested entities only)




(b) By URL type (all entities)




(c) By browser (all entities)

Figure A.1: Blacklisting over time (preliminary tests).

# APPENDIX B

## DETAILED BREAKDOWN OF PHISHTIME EXPERIMENTS

Table B.1 shows the detailed configuration of each PhishTime experiment, as described in Section 5.6. The deployment columns show the specific experiments that were included in each deployment.

| Experiment | Total Batches | Phishing Site URLs | Phishing Site Evasion Technique | Reported (Inert) URLs | Inert Evasion Technique | Where Reported | Site Appearance | Deployment P 03/19 | 1 05/19 | 2 07/19 | 3 08/19 | 4 09/19 | 5 10/19 | 6 12/19 | Total Sites | Total URLs |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **A (Baseline)** | 6 (+1 preliminary) | 54 new .com domains (w/ randomized path) | None | Same as Phishing URLs | N/A | Multiple (APWG, PayPal, GSB API, NetCraft, PhishTank, US CERT) | paypal.com | X | X | X | X | X | X | X | 54 * 6 (+ 54 preliminary) | 54 * 6 (+ 54 preliminary) |
| **B (Basic Evasion)** | 3 (+1 preliminary) | 54 new .com domains | JavaScript cloaking: content rendered in JS | Same as Phishing URLs | N/A | Multiple | paypal.com | X | | X | | X | | X | 54 * 3 (+ 54 preliminary) | 54 * 3 (+ 54 preliminary) |
| | 3 (+1 preliminary) | 54 new .com domains | User agent cloaking: Mobile devices | | | | | X | | X | | X | | X | 54 * 3 (+ 54 preliminary) | 54 * 3 (+ 54 preliminary) |
| **C (Typical Evasion: Redirection)** | 2 (+1 preliminary) | 54 new .com domains | JavaScript cloaking: content rendered in JS | 54 bit.ly redirection links | None | Multiple | paypal.com | X | X | | X | | | | 54 * 2 (+ 54 preliminary) | 54 * 2 (+ 54 preliminary) |
| | 2 | 54 new .com domains | User agent cloaking: Mobile devices | 54 bit.ly redirection links | | | | | | X | | X | | | 54 * 2 | 108 * 2 |
| | 2 (+1 preliminary) | 54 new .com domains | JavaScript cloaking: content rendered in JS | 54 new .com domains | | | | X | X | | X | | | | 54 * 2 (+ 54 preliminary) | 108 * 2 (+ 108 preliminary) |
| | 2 | 54 new .com domains | User agent cloaking: Mobile devices | 54 new .com domains | | | | | | | | | | | 54 * 2 | 108 * 2 |
| | 2 (+1 preliminary) | 54 new .com domains | JavaScript cloaking: content rendered in JS | 54 new .com domains | | | | X | X | | X | | | | 54 * 2 (+ 54 preliminary) | 108 * 2 (+ 108 preliminary) |
| | 2 | 54 re-used .com domains (from Experiment C, with new path) | User agent cloaking: Mobile devices | 54 bit.ly redirection links | | | | | | | | | | | 54 * 2 | 108 * 2 |
| **D (Domain Re-use)** | 2 (+1 preliminary) | 54 re-used .com domains | JavaScript cloaking: content rendered in JS | 54 bit.ly redirection links | None | Multiple | paypal.com | | X | | X | | | | 54 * 2 (+ 54 preliminary) | 108 * 2 (+ 108 preliminary) |
| | 2 | 54 re-used .com domains | User agent cloaking: Mobile devices | 54 re-used .com domains | | | | X | | X | | X | | | 54 * 2 | 108 * 2 |
| **Note: same configuration as Experiment C, except for URL paths** | 2 (+1 preliminary) | 54 re-used .com domains | JavaScript cloaking: content rendered in JS | 54 re-used .com domains | .htaccess cloaking | | | X | X | | X | | | | 54 * 2 (+ 54 preliminary) | 108 * 2 (+ 108 preliminary) |
| | 2 | 54 re-used .com domains | User agent cloaking: Mobile devices | 54 re-used .com domains | | | | | | X | | X | | | 54 * 2 | 108 * 2 |
| **E (Discovery)** | 2 (+1 preliminary) | 54 new .com domains | None | | | PayPal | paypal.com | X | X | | X | | | | 54 * 2 (+ 54 preliminary) | 54 * 2 (+ 54 preliminary) |
| | 2 | 54 new .com domains | None | | | APWG | | | | X | | X | | | 54 * 2 | 54 * 2 |
| | 2 (+1 preliminary) | 54 new .com domains | JavaScript cloaking: content rendered in JS | Same as Phishing URLs | N/A | PayPal | | X | X | | X | | | | 54 * 2 (+ 54 preliminary) | 54 * 2 (+ 54 preliminary) |
| | 2 | 54 new .com domains | User agent cloaking: Mobile devices | | | APWG | | | | X | | X | | | 54 * 2 | 54 * 2 |
| | 1 | 54 new .com domains | .htaccess cloaking (baseline) | | | | | | | | | | X | | 54 | 54 |
| | 1 | 54 new .com domains | JavaScript cloaking: Mouse movement, .htaccess cloaking | | | | | | | | | | X | | 54 | 54 |
| | 1 | 54 new .com domains | JavaScript cloaking: CAPTCHA, .htaccess cloaking | | | | | | | | | | X | | 54 | 54 |
| | 1 | 54 new .com domains | JavaScript cloaking: Popup alert, .htaccess cloaking | | | | | | | | | | X | | 54 | 54 |
| **F (Emerging Evasion)** | 1 | 54 new .com domains | JavaScript cloaking: Mouse movement | Same as Phishing URLs | N/A | Multiple | paypal.com | | | | | | X | | 54 | 54 |
| | 1 | 54 new .com domains | JavaScript cloaking: CAPTCHA | | | | | | | | | | X | | 54 | 54 |
| | 1 | 54 new .com domains | JavaScript cloaking: Popup alert | | | | | | | | | | X | | 54 | 54 |
| **G (Evidence-based Reporting Protocols)** | 1 | 54 new .com domains | IP cloaking | Same as Phishing URLs | N/A | Chrome Suspicious Site Reporter | paypal.com | | | | | | | X | 54 | 54 |
| | 1 | 54 new .com domains | US geolocation | | N/A | GSB (manually) | | | | | | | | X | 54 | 54 |

Table B.1: A detailed breakdown of each PhishTime experiment and deployment.