Detection, Triage, and Attribution of PII Phishing Sites

Dennis Roellke

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy
under the Executive Committee
of the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2023

Abstract

Detection, Triage, and Attribution of PII Phishing Sites

Dennis Roellke

Stolen personally identifiable information (PII) can be abused to perform a multitude of crimes in the victim's name. For instance, credit card information can be used in drug business, Social Security Numbers and health ID's can be used in insurance fraud, and passport data can be used for human trafficking or in terrorism. Even Information typically considered publicly available (e.g. name, birthday, phone number, etc.) can be used for unauthorized registration of services and generation of new accounts using the victim's identity (unauthorized account creation). Accordingly, modern phishing campaigns have outlived the goal of account takeover and are trending towards more sophisticated goals. While criminal investigations in the real world evolved over centuries, digital forensics is only a few decades into the art. In digital forensics, threat analysts have pioneered the field of enhanced attribution — a study of threat intelligence that aims to find a link between attacks and attackers. Their findings provide valuable information for investigators, ultimately bolster takedown efforts and help determine the proper course of legal action. Despite an overwhelming offer of security solutions today suggesting great threat analysis capabilities, vendors only share attack signatures and additional intelligence remains locked into the vendor's ecosystem. Victims often hesitate to disclose attacks, fearing reputation damage and the accidental revealing of intellectual property. This phenomenon limits the availability of postmortem analysis from real-world attacks and often forces third-party investigators, like government agencies, to mine their own data. In the absence of industry data, it can be promising

to actively infiltrate fraudsters in an independent sting operation. Intuitively, undercover agents can be used to monitor online markets for illegal offerings and another common industry practice is to trap attackers in monitored sandboxes called honeypots. Using honeypots, investigators lure and deceive an attacker into believing an attack was successful while simultaneously studying the attacker's behavior. Insights gathered from this process allow investigators to examine the latest attack vectors, methodology, and overall trends. For either approach, investigators crave additional information about the attacker, such that they can know what to look for. In the context of phishing attacks, it has been repeatedly proposed to "shoot tracers into the cloud", by stuffing phishing sites with fake information that can later be recognized in one way or another. However, to the best of our knowledge, no existing solution can keep up with modern phishing campaigns, because they focus on credential stuffing only, while modern campaigns steal more than just user credentials — they increasingly target PII instead.

We observe that the use of HTML form input fields is a commonality among both credential stealing and identity stealing phishing sites and we propose to thoroughly evaluate this feature for the detection, triage and attribution of phishing attacks. This process includes extracting the phishing site's target PII from its HTML <label> tags, investigating how JavaScript code stylometry can be used to fingerprint a phishing site for its detection, and determining commonalities between the threat actor's personal styles. Our evaluation shows that <input> tag identifiers, and <label> tags are the most important features for this machine learning classification task, lifting the accuracy from 68% without these features to up to 92% when including them. We show that <input> tag identifiers and code stylometry can also be used to decide if a phishing site uses cloaking. Then we propose to build the first denial-of-phishing engine (DOPE) that handles all phishing; both Credential Stealing and PII theft. DOPE analyzes HTML <label> tags to learn which information to provide, and we craft this information in a believable manner, meaning that it can be expected to pass credibility tests by the phisher.

Table of Contents

List of Figures

List of Tables

Dedication

To my mother and my father. You have always served as role models and led by grand example. You taught me morals and critical thinking, work ethics and the value of passion. Thank you for prioritizing my education and providing me with support and direction when needed. Thank you for letting me live independently otherwise.

To my principal advisors over the years. You taught me different approaches to academia and mentored me in distinct ways. Thank you for teaching me about life outside my comfort zone and about myself.

To my wife, relentlessly by my side, providing emotional balance, and opening my eyes to the beauty of nature.

Preface
History of Internet Crime

Today, over half of the world's population has access to the Internet and with the exception of Columbia University alumnus Warren Buffet, nine out of ten of the wealthiest people alive have made their fortune from software products. At this stage, people might find it hard to imagine daily life without access to the plentiful services available on the Internet.

0.1 Internet

With great power comes great responsibility and when a research team at the Defense Advanced Research Projects Agency (DARPA) invented the Internet in the 1960s, they soon found out that "The ARPA Computer Network is susceptible to security violations", and there is an "affection for the challenge of breaking someone's system" [1]. This affection still drives the hacking community today, presenting itself in gamified learning approaches such as Hackathons and Capture-the-Flag challenges. Computer system security quickly became a note-worthy problem for the DARPA project and its significance grew even further when in 1989 Tim Berners-Lee et al. invented and standardized the Hypertext Transfer Protocol (HTTP) protocol - a foundational step to maturing the DARPA project into the Internet we know today [2]. In 1993, over one million public hosts had Internet access and this number rose to 171 million hosts in 2003. The dot-com era is also considered the digital gold rush of the 1990s, and the movement has culminated in over one billion hosts accessible through the Internet since late 2013 [3], accounting for over 99% of all global

communications [4]. The growth and influence of the Internet show no sign of stopping, but as the scale of the Internet grew, so did the motivation for attackers to corrupt the communication. The omnipresence of the Internet on our computers and phones has made it convenient to even communicate the most sensitive information, such as health care records and financial data over the Internet. That makes it a lucrative target for criminals. As the threat rises, most companies make great efforts to protect their systems and plenty of sophisticated cyber security solutions are available today. It has become an established career path for Software Engineers to specialize as Security Engineer, Penetration Tester, or Threat Analyst and governments have legislated several laws, like the Computer Fraud and Abuse Act of 1986 (CFAA), industry-specific regulations like the Health Insurance Portability and Accountability Act of 1996 (HIPAA to protect)patient health information and the Payment Services Directive Two (PSD2) that requires strong customer authentication for certain online money transfer. In the face of enhanced protection mechanisms, criminals shift their efforts to the weakest link. They bypass corporate security and directly assault Internet users, scamming them with social engineering campaigns, often in the form of phishing. Although considered rather unsophisticated, phishing is the most common web attack today and most successful attacks use phishing as the initial step [5].

0.2 Cybercrime

Criminal activity in the online space is referred to as cybercrime or electronic crime, and just like traditional crime, it is conducted by individuals, criminal organizations, and even nation-states [6]. Compared to traditional crime, cybercrime mitigation is uniquely challenging. Entrance barriers are low since no physical tools are required other than an Internet connection and an attack may be direct against anyone on the Internet, across the entire world. Simultaneously, criminals have increasing access to anonymization techniques to cover their tracks and underground economies are growing, providing malicious tools to anyone mal-intended, not matter how technically unskilled [7]. The fact that cybercrime can be conducted by anyone and from anywhere leads to the rapid adaption of technology trends. In addition to that, information theft on the Internet has

the unique characteristic that criminals steal copies of the data, meaning that the data cannot be noticed as missing.

Legal prosecution of cybercrime is highly complicated because solid evidence is often either entirely absent or spread across multiple non-collaborative institutions such as Internet Service Providers (ISP), VPN providers, web hosters, underground forums, or legitimate Software-as-a-Service providers that have to value their user's privacy unless proven guilty. To further complicate this situation, evidence may be subject to a different jurisdiction if it accrued in different states or nations with different rules, or such who do not collaborate with each other. In recent years, we saw a number of prominent convictions of distinguished cybercriminals behind big hacks, but the situation has not improved with regard to small attacks against a very large number of victims. It is important to understand that everyday cybercrime does not necessarily make headlines, as it often involves smaller attacks that the victims do not bother to report, or the victims feel pitty for being a victim. The frequency of smaller attacks makes them collectively no less significant than high-profile threat campaigns. As part of their campaigns, criminals host thousands of fake websites each day to trick their victims into making payments or revealing sensitive information [8]. The variety of different schemes observed [9] suggests a high number of unrecorded threats and a variety of techniques that are difficult to predict and prevent due to their complexity. The complexity, number of stakeholders and international distribution of attacks also makes it difficult to measure the damage caused by cybercrime. Many researchers agree that the damage of cybercrime must be under-reported by victims and underestimated by governments [10].

In late 2020, security vendor McAfee estimated annual global damage due to cybercrime to cross the $1 trillion dollar mark for the first time ever. Over 1% of the total GDP of the world's developed economies, and a stark increase over the last years. Recent incidents with broad real-world impact illustrate the increasing consequences of cybercrime crossing the boundaries of computer networks. In 2017, "Half of All Americans" [11] detailed financial information was stolen in a single unprecedented breach of Equifax, an American credit scoring company. The estimated $1 billion of direct damage to Equifax are negligible compared to the long-term consequences

like irreversible brand damage, espionage [11] and mass identity theft. A different type of attack known as ransomware made headlines around the same time. The encryption-trojan *WannaCry* took hostage of over 300,000 computer systems worldwide and caused real-world damage by disrupting hospitals, production factories, and public transportation networks [12].

Ultimately, criminals lurk everywhere on the Internet, causing cause damages in terms of time, money, information, or intellectual property. As one could expect, a variety 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 have to make a huge effort to defeat each risk. Defense efforts are caught in a cat-and-mouse game trying to maintain a proper understanding of attacks and implementing sufficient mitigation to keep up with the attackers sophistication.

0.3 Phishing

In phishing campaigns, fraudsters use social engineering to trick their victims into sharing sensitive information. Here, we loosely distinguish between spear-phishing, which is targeted toward certain user groups, whale-phishing, which targets high net-worth individuals, and lastly, the general untargeted phishing attacks against general Internet users. The latter attacks can be thought of as spray-and-pray attacks that strategically value the quantity of careless Internet users over the quality of a specific target. These undirected, campaigns can be implemented easily and are considered to be operated by script-kiddies, who use easy-to-use phishing kits to orchestrate their attacks. This type of phishing is a prime example of the everyday cybercrime we mentioned in the preface: It has a low entrance barrier, results in smaller damages, but is found at a large scale. At first glance, a spray-and-pray campaign may seem trivial, and one might falsely assume that protecting against them does not require much sophistication. This assumption shows to be wrong since the threat exists until today and in spite of many solutions have been proposed over the years. Early solutions, like individual security toolbars for the browser [13] have been replaced by collaborative systems natively available in modern browsers [14, 15, 16]. Similarly, most email providers

protect their users with solid security suits and companies invest in anti-phishing solutions like user awareness training [17] and two-factor authentication [18]. Despite all efforts the number of phishing attacks continues to increase and the campaigns evolve in sophistication. In response, the academic community started to collaboratively face the challenges of "phishing, crimeware, and e-mail spoofing" by founding the Anti-Phishing Working Group (APWG), a platform for businesses, organizations, and governments to share experiences and expertise in defeating the distributed problem. The APWG shares phishing intelligence and regularly publishes trend reports on phishing. In an original report from October 2004 the APWG counted 1,142 unique phishing websites and 6,597 unique phishing e-mails [19]. Today, the numbers have surged to hundreds of thousands unique instances per month [8, 20]. The Google Safe Browsing initiative's Transparency Report presents phishing to be at an all-time high, and concludes that phishing has almost completely replaced malware as the primary threat against web browsers [21]. Monetary losses to phishing alone cumulated up to $6.9 billion in 2021 [22].

Both commercial and independent anti-phishing reports naturally survey an ever evolving subject and adjust and improve the ways in which metrics like the number of phishing domains are gathered. Although this degree of change we observe from one report to another makes the reports little reliable, we can agree that the amount of phishing attacks is not becoming any less. Longterm, the ratio of phishing websites to all Internet hostnames has more than doubled rising from approximately 0.047% in 2005 to 0.102% in 2018 [23]. Subsequently, the anti-phishing community generally agrees that despite all efforts phishers are still making a lucrative return on investment to justify growing their campaigns.

The persistent threat posed by phishing is both cause and effect of many difficult challenges faced when trying to protect victims, who are often unaware and defenseless. Although a plentiful legislature is in place to outlaw phishing campaigns and hackers as criminal organizations, the technical challenges of detection, triage and attribution have yet to be solved. Finding technical countermeasures against the distributed, unpredictable threat has become the key to user safety at scale. Any such technical solution must consider that phishing attacks tend to leverage the targeted

human's carelessness or lack of attention as the weakest link. For example, human targets may find themselves ignoring even the most expressive anti-phishing systems if their warnings frequently raise false alerts and become an annoyance (the threshold for which is individual and context dependent).

In this game, phishers are generally at an advantage because their success does not depend on every single target, whereas defenders aim to defeat every single attack. Additionally, the wide availability of exploitable Internet infrastructure and the lack of universal authentication on the internet enables these attacks to be performed at a large-scale low cost.

It is commonly understood that organizations impersonated by phishers have a crucial role in this duel, basically facing (corporate-) identity theft and unwillingly providing the battleground for this attack-defense game.

Chapter 1: Introduction And Background

1.1 Thesis Statement

The fight against online crime is an ongoing cat and mouse game and the incentive for online fraud rises with the ongoing expansion of the Internet. Particularly phishing attacks are easy to execute by using phishing kits and the incentive for the crime scales with an increasing number of Internet users. The attacker's use of independent services distributed across the Internet complicates the mitigation of phishing, such that today, lowering the phishers' incentive is the best we can do. Any data point about the criminals can help to understand their operations, and learn how to stop the malicious efforts — ideally prosecuting the individuals behind it. We hypothesize that spray-and-pray phishing campaigns are created with phishing kits that leave an implicit mark on the sites. Furthermore, we observe that security solutions like Multi-Factor Authentication raised the bar for Credential Theft by protecting the user's passwords. Similarly, recent payment network regulations were established to raise the bar for fraud on financial transactions. We hypothesize that criminals today found ways to directly bypass these mitigations and indirectly bypass them by shifting their focus from these increasingly well-protected assets to another asset, which is at least as valuable: Personally Identifiable Information (PII). We propose to measure this shift and to quantify a phishing site by the PII it steals. Finally, we claim that active infiltration is a suitable way to study crime operations and that our analysis provides a promising foundation to automate decoy deployment. Properly crafted decoys can trick phishers into believing they were victim information and the data can then be traced throughout the Internet. Our case study provides a reference of which decoy information phishers reuse, when, and where. This knowledge provides evidence of abuse and coordinate between the affected parties, e.g. government institutions, merchants, financial service providers, and internet service providers. We examine the following hypothesis

Hypothesis *Phishing is s shifting from Credential Theft to PII Phishing. PII Phishing campaigns are created with phishing kits that leave an implicit mark on the sites. The mark can be used for phishing classification and threat actor fingerprinting. Properly crafted decoys can trick PII Phishers into believing they are victim information and the data can then be traced throughout the Internet.*

1.2 Contributions

The studies presented in this thesis provide the following contributions to the academic community:

- A five year dataset of raw phishing websites including the page source
- The feature importance of input field identifiers as a new phishing site feature
- The feature importance of input field descriptions as a new phishing site feature
- Application of code stylometry to de-anonymize phishers
- A new method to categorize phishing sites into three distinct threat levels
- Study PII Phishing over five consecutive years
- Study the distribution of natural languages in phishing campaigns
- A PII reference list
- A new method to profile phishing sites
- Reference study to show the implications of phishing kit profiling
- A new method to draw links between phishing sites
- A system to automatically generate believable fake input data for phishing sites
- Evaluation of the relevance for decoy data to be *believable*

- Trace the phisher's use of fake input data through the internet
- Reference study of phishers' attempted reuse of decoy data

1.3 Background

The preface contextualized phishing as an instance of cybercrime. We discussed the historic evolution of the threat, its current state, and the importance of anti-phishing programs. On today's internet, fraud is omnipresent and was estimated to cause over 44 million USD in losses in 2021 alone [22]. Prior to presenting our contributions, we will now provide the technical background for phishing attacks and we delimit phishing attacks from other fraud.

Phishing is a commonly used term, and one could argue that almost everyone in society has heard of phishing and has an intuitive understanding of what it is. However, some might think of email phishing, while others think of a phishing website and it quickly becomes clear that the term is not as narrowly defined as it could be. Therefore we will now delimit what we mean when we talk about phishing. The next section covers the work of a phisher and the section thereafter covers the work of a threat analyst. Figure 1.1 illustrates how their operations.

1.3.1 Phishing Deployment and Distribution

On a high level, the initiator of a phishing campaign is concerned with two main steps, the deployment of the phishing site on the web, and the distribution of a link to that website. In a first step, the phisher prepares a web server, which can either be hosted through public service, or it can be someone else's web server that the phisher compromised by exploiting a vulnerability on that system. E.g. vulnerability scanners provide straightforward ways to succeed with this step [24]. On the webserver, the phisher then clones a target site and rehosts it, mimicking the original brands' appearance, but being connected to the phisher's backend instead of the original platform. In the meantime, the phisher registers a domain name that points to that phishing site and distributes a hyperlink to that website via email or other distribution channels, such as text messages. Note that every phisher has mutiple options within this framework, such as choosing different distribution

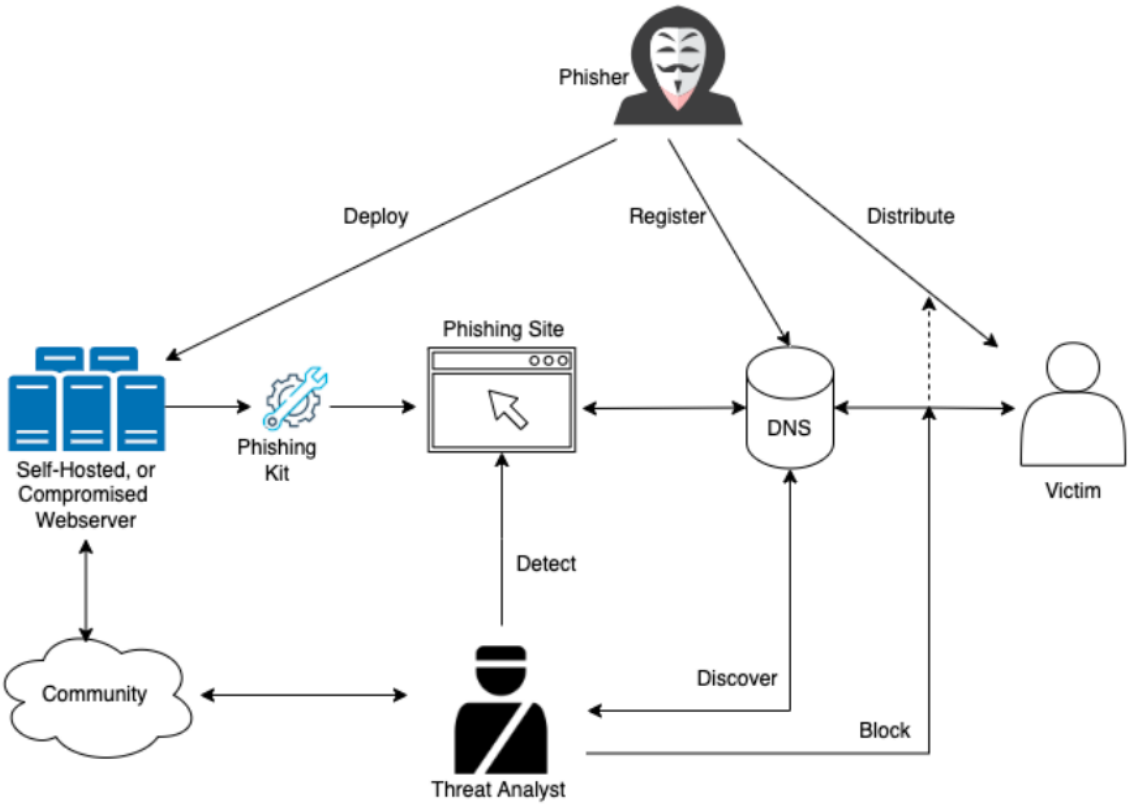


Figure 1.1: Deployment, distribution, discovery and defense of phishing.

channels or not registering a domain name, using e.g. the compromised website's domain or the server's IP address directly. Different strategies may be more or less successful to evade discovery by the anti-phishing community as we will see in the next section.

1.3.2 Phishing Discovery and Detection

On the other side of the table, a threat analyst tries to discover as many phishing sites as possible and confirm whether they are phishing or not. Once it is confirmed the analyst publishes the phishing URL on a blocklist, which will block future victims from visiting the dangerous site. In particular, the threat analyst takes on the tasks of discovery and detection, meaning the confirmation and blocking.

The discovery is often done using DNSTwist, a generator for suggestive domain names that look like real brand names. For example, using a polygraph of the site *paypal.com* with the digit

1 instead of the letter l - *paypa1.com*, which looks deceptively similar - or typo-squatting domains like YouTub.com instead of YouTube.com. Another discovery method is to monitor domain name registration, but with over 250,000 [1] domains registered per day, this is a mammoth task that only a few companies approach [25]. A more efficient method is to monitor unexpected spikes in traffic to a site, because if a formerly unknown site sees a high access rate it is fair to ask where this sudden popularity comes from and if it may be a scam. While these methods depend on domain names, not every phishing site has a domain name associated with it. Especially if the phisher decides to host the site on a compromised server, the domain name will still point to the benign website. In this case, the malicious landing page is commonly dropped in a subdirectory using polygraphs for the directory name rather than the domain name itself. These cases are generally reported by a victim or by the victim's web server itself. When a URL is reported as potential phishing it is the threat analyst's task to confirm the accusation in order to avoid harming benign websites with false-positive blocklisting. Unfortunately, the classification of community reports is inherently retroactive, because it can only be done after there was a first victim who reported it.

After the detection, URLs are shared with blocklists, lists that prevent the user from visiting the blocklisted site and showing a warning instead. We differentiate between three types of blocklists: In-browser blocklists, most prominently represented by Google Safe Browsing in Chrome, Safari, and Firefox, or Microsoft Smart Screen in the Edge Browser. Commercial solutions like APWG's eCrime Exchange are often found in corporate intrusion detection systems and open source solutions like OpenPhish and PhishTank are frequently used in research experiments. Section 2.2 debates the importance of a unique agreed-upon dataset. Another approach to blocklists is to block the distribution of the site's hyperlink. These solutions are often integrated into corporate intrusion detection systems rather than the browser because most of them leverage the access to the mail server and network traffic which they leverage to recognize patterns. We see a positive development in several services' integration of free solutions like email providers now often run phishing filters for their customers. The details of phishing email mitigation are out of scope for this work,

[1] note that a similar amount is deleted every day

in which we focus on phishing websites only.

A careful review of each of the threat analysts' steps reveals several limitations. Any discovery apart from domain registration monitoring is inherently incomplete, and even domain registration monitoring suffers from the phisher's ability to change the website's content on-the-fly. In other words, a crawler might classify the newly discovered domain as "benign", but later on the phisher can swap it with a malicious site.

The detection, meaning the confirmation and blocking, of phishing sites is rather slow, taking about one hour from reporting it to getting it on the blocklist. Research shows that there is a strong bias for certain sites and non-commercial sites like government websites are generally less well detected. On average, 30-60% of sites bypass detection through the use of cloaking. Distribution vector blocking is much more successful, but is still estimated to oversee 5%. Of those 5%, approximately 10% of recipients eventually click the link.

At the bottom line, deployment and distribution of phishing sites are straightforward operations, but the discovery and detection require great efforts of coordination between multiple parties. The ease of deployment and distribution is further facilitated by the availability of so-called phishing kits, which we describe in the next section.

1.3.3 Phishing Kits

A phishing kit is an abstract concept, describing the toolboxes used for the deployment of a site on a server. The complexity of these kits varies widely and while some phishing kits offer an administration panel to manage spam message distribution others offer unhindered access to readily compromised web servers. In this work, we focus on kits that help with the hosting of the phishing website itself. They may integrate custom phishing sites, provide phishing site templates, or create clones of a target brand [82]. Crucially though, they also implement the backend to handle the data dump for the fraudster. They may for instance store the stolen user data in a text file or directly transmit it to the phisher via email. Furthermore, it is common phishing kit practice to automatically move the page source from one subdirectory to another, effectively changing the

URL and thereby bypassing blocklists. Although we can expect some phishers to use their self-written kits themselves, it is important to understand that the creation and distribution of phishing kits is a business in and of itself, where some criminals even offer custom feature requests for the kit [12] or so-called phishing-as-a-service (PhaaS) solutions [74].

Phishing kit creators compete in underground markets where they get ranked based on the effectiveness, ease of use, or perceived security (i.e., from anti-phishing systems) of their kits. All in all, easy access to these tools lowers the barrier to entry and allows minimally skilled criminals to become successful phishers. In fact, these tools even remove the need to speak the target language and they can often be deployed multilingual and across borders. This ease of access has primed the term script-kiddie, a kid (the minimally skilled criminal), who uses a script (the phishing kit) without fully understanding the details of the script. A script-kiddie just purchases the kit online, enters the destination email address to send the stolen information to, and uploads the generated code to the readily available web server. Down to the spam email distribution, every step of the way can be fully managed for the scammer, so she can lean back and wait for stolen credentials to fly into the mailbox.

The heavy sharing of phishing kits in underground communities is the foundational observation that motivates and enables our phishing detection research in Chaper 2.4.5 Section 3.1 and the attribution of a phishing landing page to a phishing kit in Chaper 2.4.5 Section 3.3.

1.3.4 PII Phishing

Online fraud is an ever-growing problem causing billions of dollars of losses. While the fraud is generally motivated by financial opportunity behind the scam schemes, we distinguish between trick scams and identity theft. Trick scams like fake products, or ransomware are solely financially motivated and usually end with a transaction. Identity theft on the other hand extends the threat from financial harm to an impersonation of the victim. When compared to trick scams, identity theft may also target the victim's finances, but the real impact often remains unknown. E.g., victims' bank accounts may be used as mule accounts, their address for drug trafficking, or their

work account for industrial espionage. All three of the above are the result of a so-called Account Takeover. In most cases, an account can be taken over by abusing the username and password combinations obtained from phishing. However, Account Takeover is becoming increasingly difficult for hackers, who are now facing the wide adoption of Multi-Factor Authentication and the FIDO standards. It has been shown that Man-in-the-Middle attacks can overcome these protection mechanisms, but they reasonably lift the bar. Simultaneously, we observe an increasing theft of personally identifiable information (PII), discussed in Chapter 4. This trend suggests the new risk is Unauthorized Account Creation (UAC). Our case study in Chapter 10 shows that cybercriminals today, leverage stolen PII to create fake accounts in the victim's name. We propose a system to infiltrate the hackers and learn more about their intentions. At this point, it is not clear what UAC is being used for.

1.3.5 Summary

As technology advances more and more people get access today and today, most of the developed world is on the Internet around the clock. An ever-increasing number of Internet users provide an ever-increasing number of victims for cybercriminals and while technology advances so do the tooling for fraudsters. The simplicity in which phishing occurs at the first glance may falsely suggest that it is straightforward to mitigate it, or one might even expect it to be a solved problem today. In reality, though, the threat is rising. It is the volume and the diversity of campaigns that make it difficult to discover or detect them and on the other hand, it is alarmingly easy for criminals to engage in phishing campaigns without any prior technical knowledge.

In the next chapters, we study the use of two new features of phishing websites for the Detection, Triage, and Attribution of phishing campaigns. We unveil the trend of PII phishing and we propose to fight back by interacting with phishers through decoy-data-based sting operations.

Chapter 2: Technical Approach

In the previous chapter, we have learned that phishing is a social engineering technique that is easily accessible even for criminals with comparatively low technical sophistication. Any criminal mind can run phishing campaigns by using easily accessible phishing kits that automatically deploy malicious websites and send the access link to potential victims. We observe that many phishing kits leave an implicit mark on the website they create. This chapter introduces this "mark" as new feature set to train machine learning algorithms on. Different combinations of the two new features are particularly well suited to detect if a site is a phishing site, to triage if it is a PII phishing site and to attribute the site to a known set of phishing kits. We motivate this use with a case study in Section 2.4.

2.1 Features Engineering for Phishing Classification

Even though the technical depth of the threat may seem shallow, the threat is real and its true danger lies in the low entry barrier yielding a large quantity of small independent attacks from distributed origins and targeting independent victims. The anti-phishing landscape has matured and we see native implementations of free blocklists in almost all modern web browsers today, but we also exhibit room for improvement when it comes to proactive phishing detection and the quantification of a phishing site. This chapter will introduce two new features to detect and triage phishing sites by. The features can be extracted from any given (phishing) website and they can be used in any given machine learning algorithm. Next, this section delimits our approach from existing approaches in the motivation section, Section 2.1.1, that includes related work. Then, we present the details of our method (see Section 2.1.2) and the case study in Section 2.4 examine its implications. As a result, we will see that in addition to the classification of whether a site is phishing or not, and what is being phished for, the features are well suited to perform a phish-

ing kit attribution of each site. Chapter 2.4.5 considers the application to Detection, Triage, and Attribution of PII Phishing attacks.

2.1.1 Motivation

Existing anti-phishing solutions either implement the detection of content similarity or behavioral similarity between a phishing site and a brand. Behavioral similarity analyzes patterns in the data by leveraging access to meta-information about the attack. It is therefore well suited for use in Security Operation Centers (SOC), but the general public does not have access to such enterprise resources when surfing the Internet from personal devices. Hence, we consider behavioral solutions outside of the scope of this work and focus on the presentation of content similarity-based solutions. Historically, various hash comparisons of all source files in a web project have been proposed as an intuitive solution [26] and they have been extended to URL segment comparison [26]. Since then, smarter solutions have investigated the high-level "style" of websites using techniques such as HTML tag enumeration, word count, and image profiling [27, 28, 29]. Today's cutting-edge solution in detection research is a 16 layer convolutional neural network called VisualPhishNet that classifies phishing sites based on screenshots. VisualPhishNet is the result of a long line of AI and Deep Learning applications to the phishing task, but most of which assumes prior knowledge about which site the phisher is cloning, e.g., the classifier asks: "Is x a clone of y given y?". Furthermore, it has been shown that even VisualPhishNet's high accuracy of 89% can be can be decreased to 69% through adversarial examples [30]. A different line of work considers the de-anonymization of programmers. Intuitively, an application of de-anonymazation techniques to phishing seems intriguing. A paper by Caliskan et al. attributed C/C++ source code to one of 250 programmers based on their implicit preferences for layout, lexical, and syntax when writing code [31, 32]. They used a random forest model and achieved 98% accuracy, but the approach has not been applied to web applications. By generalizing a hacker's programming style, one may be able to detect entirely new cases, zero-days, that are cloning formerly unsupported brands. We follow this work's intuition and analyze the criminal web developer's programming preferences,

asking where do phishing sites differ from benign sites?

2.1.2 Form Field Labels and Identifiers

The technology stack of a phishing site is identical to that of a benign website. While the backend may be arbitrarily complex, frontend technologies are standardized by the World Wide Web Consortium (W3C) and they mainly use HTML, CSS, and JavaScript. Especially if we assume that a phishing site is a direct clone of a benign target brand most of its page source components will be identical, or at least similar. We know that HTML anchor tags have been studied at length and many phishing classifiers use the ratio between external and internal links as a feature. The number of image tags in a website is another popular phishing classification feature. Carefully dissecting page source components, we observe that more features exist, that have no yet been investigated in the academic literature to the best of our knowledge. Given that it is a phishing sites primary purpose to catch victim data almost every phishing site implements a form field to enable data submission. Form fields commonly use the HTML <input> tag, which we will now study in detail.

According to the HTML language specifications, each <input> tag on a website can be uniquely

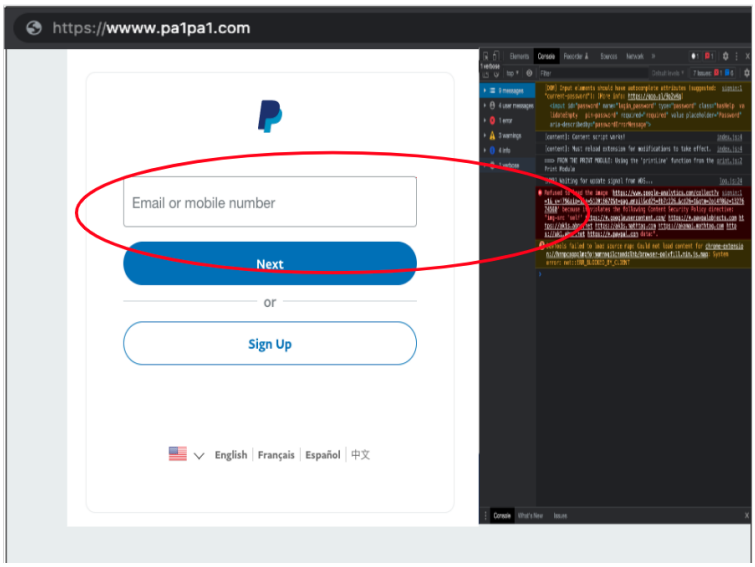


Figure 2.1: Most if not all phishing sites implement form fields.

identified by its ID attribute. Other tags can therefore use the ID attribute to refer to an input field. For example, the <label> tag has a an attribute called *for* to maintain which input field it is labelling. It provides a description for the user to know which information to enter into which field of the form. According to the HTML5 standard, we can find user-facing label information at five distinct locations [33], highlighted in Listing 2.1.

```
1  <input type='email' label='Email'>
2  <input type='email' *-label='Email'>
3  <input type='email' placeholder='Email'>
4  <label for='asd'> Email </label> <input type='email' id='asd'>
5  <div> Email <input type='email'> </div>
```

Listing 2.1: Standard implementations of HTML labels to query users for specific pieces of information.

Similar to the <label> tag, other web technologies, most prominently JavaScript, can access HTML components by their ID. For example, it is a common practice to process user input using JavaScript. In this case, JavaScript implicitly uses the input field's ID attribute as a variable name to store user input in. In doing so, the input field's ID leaks the phisher's personal preferences - the choice of a variable name. As we will show in the case study in Section 2.4 phishers modifiy these variable names for integration with the phishing kit backend, or the site may not be a clone but be generated by a phishing kit that supports the impersonation of many kits, but all using the same form fields. Similarly, to the input field ID, the input field descriptor leaks information about the phisher. Regardless of whether the descriptor is implemented as a label or a placeholder (see Listing 2.1), it summarizes what the phisher is interested in stealing .

This work proposes to use the both the HTML <input> tag's ID attribute and its descriptor as a new feature for the Detection, Triage, and Attribution PII Phishing Campaigns. This approach is promising under the hypothesis that even if most of a phishing site's page source is copied from a target brand, fraudsters modify the input fields for their individual needs. Chapter 2.4.5 Section 3.1 discusses how we use a phisher's personal preferences, like the ID attribute, for phishing detection. Chapter 2.4.5 Section 3.2 shows how we parse input field descriptors to triage a phishing site as one of three threat categories, Contact Gathering, Credential Theft or PII phishing, and Chapter 2.4.5 Section 3.3 promotes the how we can use the features to profile a phishing site and build clusters

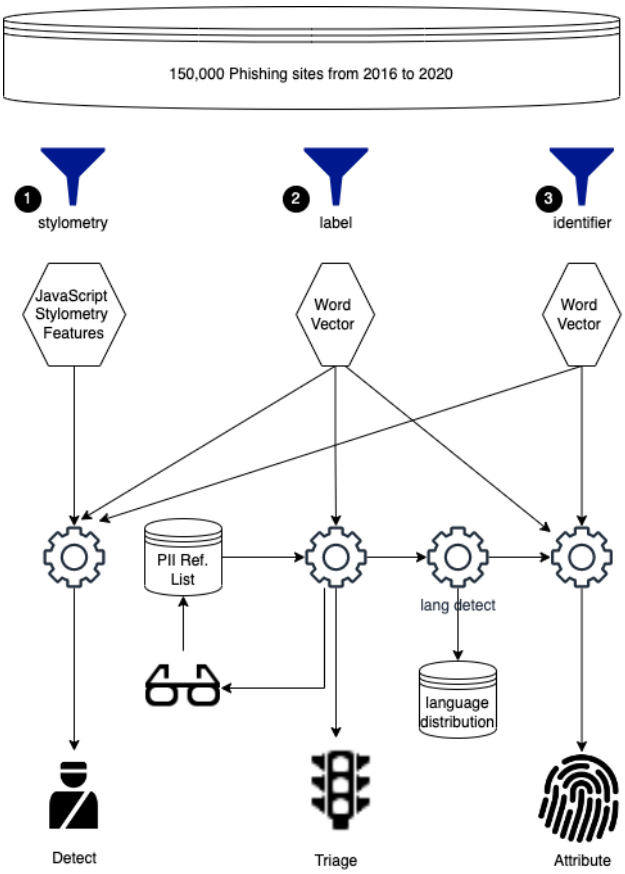


Figure 2.2: Overview

of related phishing kits.

Our analysis engine is designed to dynamically execute phishing websites, such that on top of the site's static HTML all JavaScript-generated content modifications are rendered. This way, it is ensured that we analyze exactly what a potential victim is presented with. At first, we parse the page source to extract the lexical, layout and syntax features from Caliskan et al. [31] that we introduced in the previous section ❶. Then, we extract input field descriptors ❷ , and input field identifiers according to Listing 2.1 ❸. We then use respective combinations of these features to Detect, Triage and Attribute PII Phishing Campaigns as depicted in Figure 2.2

First, the detection engine uses all three feature classes. The evaluation in Chapter 2.4.5 Section 3.1 highlights that stylometry features cannot decide the task by themselves, but using label values and input field ID attributes raises the accuracy from 68% to 92%. Second, the triage system only uses an ordered word vector of label values to abstract any given website. For example,

a generic landing page that asks for the visitor's email address and account password will be abstracted as the vector <email, password>.

The word vector is then used to identify the personal or sensitive information that the phisher is trying to steal. Each element in the word vector indicates a threat category. Our engine facilitates this mapping and ranks the site with respect to its most severe threat category. Details in 2.4.5 Section 3.2. As a side-effect, this process iteratively refines a PII reference list that we discuss further in Chapter 2.4.5 Section 3.2. Furthermore, label vectors can be used for language detection, revealing the target audience of the scam. The third piece of our system fingerprints phishing sites for threat actor attribution in Chapter 2.4.5 Section 3.3. Here, attribute phishing sites to phishing kits they may have been created with. In the absence of a ground truth dataset with labeled phishing kits, we propose to use the CorpRank algorithm to build a hierarchical social network of phishing kits. This approach is based on the observation that JavaScript and PHP internally use the HTML <input> tag ID attributes as variable names. In other words, a phisher, or phishing kit needs to specify them to send the stolen information from the frontend to the backend of the site. As a consequence, these development artifacts may hint at the phishing kit that generated, used, and reused them. The presence of identical variable names between two or more websites indicates a relationship between the toolkits that generated the site.

The next section presents the dataset that we use for the rest of this project. After that, we illustrate the empirical results of a case study in which we manually reviewed the implications of this feature. Chapter 2.4.5 Section 3.3 further elaborates on the topic of phishing kit fingerprinting and shows how mapping them into a social network allows us to quantify relationships between phishers.

2.2 Dataset

Along with new methods for employee training [34, 35, 36, 37, 38], life cycle analyses [39], and case studies [40], many new techniques to detect phishing have been proposed in the last years [41, 42, 43, 44, 45, 46, 47, 48, 49]. We found that the attempt to compare these techniques

unveils a structural limitation in state-of-the-art phishing research: the lack of a uniform, agreed upon, and consistent benchmark dataset. Related work suggests the availability of many different phishing datasets, but most of them are blunt collections of tabular index data. They fall short by not providing raw data. Recently, the PhishBench project introduced a framework to compare new machine learning classifiers [50]. While this is a tremendous contribution, PhishBench is only helpful for model comparison, and less so for developing solutions that go beyond the choice of the model. For example, PhishBench uses a fixed set of already extracted features, which is restrictive since without raw data we cannot research new features. We present a new source code-based feature that differs from existing content-based features like the frequency of internal and external hyperlinks. Similarly, though it has been repeatedly reported which industries are targeted, raw data allows us to ask additional questions, such as what information has been stolen. A common alternative to a benchmark dataset is to leverage publicly accessible feeds like PhishTank and OpenPhish. In fact, Chiew et al. compare 38 phishing papers, 36 of which evaluate their systems on PhishTank (two rely on OpenPhish and Google Safe Browsing) [51]. While we agree that this is a good solution, we note that the feeds themselves are error-prone and their sources are not reported, hence, even with raw data from phishing feeds, the ground truth may be skewed[52]. Furthermore, any point in time evaluation may be subject to seasonal biases like the difference between phishing hosted around Christmas compared to tax season[53]. Lastly, we know the indexing of phishing feeds to be delayed by several hours, meaning that the data can only reflect sites from a later life cycle phase, and cannot be used to study an attacker's initial activity profile. Once a site is listed by a phishing feed, it is likely that the phisher has already abandoned the project [39]. Researchers developing new phishing classifiers often overcome this hurdle by leveraging their proprietary access to company data and insider information. Utilizing proprietary data is invaluable for classifier development and the insights benefit the community through stronger and faster detection, but from an academic standpoint, it is crucial to facilitate an objective comparison of models - even among models that use different features. Other research branches, like the bug-finding community, have long realized this issue and they have employed a number of easy-to-use representative benchmark

Dataset Name	Collection Period	Data Source	Tool(s)	# Pages	# Features
PhishBench (2.0) [50]	9/5/2018 - 10/30/2018	PT, OP, APWG	n/a	95,524	200/**X**
UCI Phishing [57]	03/26/2015	n/a	n/a	181,559	30/**X**
PhishMonger[58]	3/15/2016 - 5/13/2016	PT	wget	88,754	*all/*
Phishing Dataset for ML: Feature Eval. [59]	5/2017 - 6/2017	PT, OP	n/a	5,000:5,000	48/**X**
UNIMAS Phishing Dataset [51]	5/2017 - 6/2017	PT	wget, webshot, whois	15,000:15,000	*all/*
BlackPhish[60]	4/2018 - 10/2018	PT	Propr. Java app	4,097:5,438	*all/*

Table 2.1: Overview of state-of-the-art phishing datasets

datasets that have become broadly adopted [54, 55, 56]. In our need for a representative dataset, we found three peer-reviewed studies that open source their raw phishing data. We combine these datasets and extend them with our own data collection. We compile them as one benchmark dataset and we open source five years' worth of raw phishing data - a bundle of 165,387 sites that meet the following criteria:

1. Visual renderability
    - Raw Client Side Source Code (HTML, CSS, JS)
    - Media and Image Files (.png, .jpg, ...)
2. Hosting Information
    - Raw URL/ Domain Name Information
    - WHOIS Records

Table 1 lists six open-source phishing datasets published in prior work.

We note that the datasets are anywhere between three and six years old and range in size from 4,097 to 181,559 phishing sites. However, the largest dataset is also the oldest, and the two largest datasets do not provide raw data. For our new dataset, we aggregate the three datasets that have raw data, indicated by a checkmark in Table 1 column Features. We then complement it with our own collection from September to October 2019 and January to February 2020.

	Original Dataset	Rendered (erroneous)	Unique URL
Benign	35,983	16,023 (19960)	11,697
2016	5,077	3,732 (1345)	3,559
2017	20,206	11,275 (8931)	3,364
2018	4,097	3,661 (436)	2,979
2019	100,004	63,638 (36366)	46,405
Total	165,367	98,329	68,004

Table 2.2: Overview of the dataset compiled for this study, showing year of collection, size, whether we were able to render it, and the availability of original domain names.

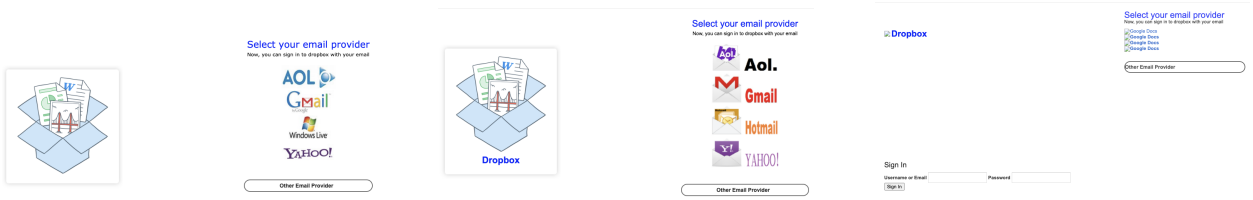
During these time periods, we scraped the OpenPhish live feed and stored non-duplicate instances of raw phishing sites using wget. We further add WHOIS entries and URLs to the data. We note that any change in the phishing landscape due to the global health crisis recognized in March 2020 is out of scope for this study [61].

While our dataset can only approximate the real phishing landscape, the aggregation of diverse collections over different time periods and varying peer-reviewed publications reduce potential biases and it does not introduce additional collection bias compared to existing collections that also use proprietary detection mechanisms or rely on public feeds. However, we provide potentially interesting historical data for longitudinal studies and time series analysis. The details of our dataset can be found in Table 2. We will now use this dataset to evaluate our approach to Triage, Detect, and Attribute PII Phishing Campaigns.

We will now share the results of an empirical study performed on this dataset and we will thereby review the implications of our newly introduced features.

2.3 Case Study

We have motivated the use of input field identifiers and input field descriptors as potential indicators for phishing. We have also reasoned that they reveal the phishers goals, and they leak a bit of information about the phisher himself, e.g., which phishing kit she prefers. To better understand the implications of the leakage, we will now present our findings from a manual review of the dataset. Mainly, we extract the new features from every site in the dataset and group the sites



(a) First instance of an obvious phishing attempt using five brands at once.(2017)

(b) Second instance of the same kit but supplied with slightly different logos. (2017)

(c) A third caption of the same kit but from a different year in the dataset (2016).

Figure 2.3: Three instances generated by the same phishing kit. Hard to detect by image processing and URL monitoring, but easily detectable by our novel technique - clustering input field IDs.

by similar feature values. Then, we randomly open a sample set of the sites and visually compare them.

While the dataset holds over 150,000 websites, it would not be very interesting to enumerate the number of kits in the dataset. Instead, we manually review cases that reflect novel insights such as the relationship between the kits. To evaluate the value of these insights for active defense and enhanced attribution one would need a playground or ground truth dataset, which we have for phishing classification, but not for underlying kit usage. Alternatively, we perform five case studies to demonstrate how the new datapoints can contribute to threat intelligence investigations.

2.4 Case Study Findings

2.4.1 Case Study 1: Variable Names as Phishing Detector

Some phishing attempts are easy to spot for the trained eye. The Example shown in Figure 2.1 shows a fake website that suggests access to Dropbox files through one of four different authentication methods, none of which is Dropbox itself. The mock-up fails to use any of the five brand logos correctly and seems intuitively suspicious. While recognizing such counterfeit is an easy task for the human eye, it may still be challenging to provide a technical solution that is able to label this site as phishing. Most recently, image classification-based approaches have been proposed to detect phishing, but an image classifier may be fooled by the far-from-real logos and it may not be able to attribute the page to a particular brand, as it incorporates five brands at once. Our solution

however abstracts the visuals away and emphasizes the fact that the three slightly different fakes in Figure 3.2 rely on identical input-field variable names:

<Emailother, Passwdgmail, Passwdother, Passwdyahoo, username, mailtype, Emailgmail, i0118, lgnId1, pwdId1, Emailyahoo, passwd, Email, Passwdhotmail, Emailhotmail, Emailaol, Passwdaol, Passwd >

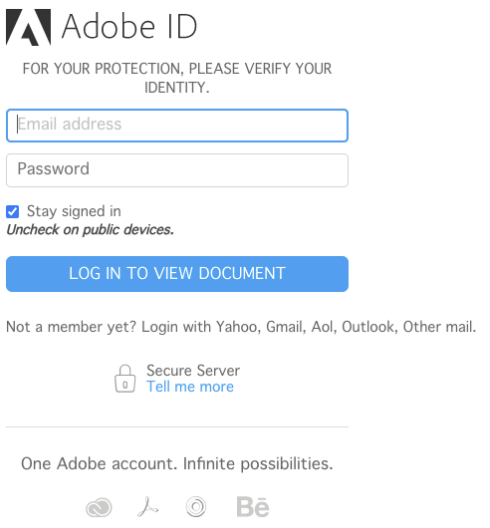
Here the same kit was used to create two visually different sites. While our 2016 data set does not contain image files (see Figure 2.3c), we can learn from it that the same kit has been active in at least 2016 and 2017. This case demonstrates our claim that input field ID analysis can be used not just to cluster sites but also to initially detect a phishing site.

2.4.2 Case Study 2: A social graph can track phisher's adaptation

We can use overlapping vector elements to build clusters, by linking vectors that (partially) include each other. For example, we find the following vector in the dataset:

<bgresponse, Passwdhidden, emr, sessionstate, SessionState, Email, checkedDomains, continue, identifiercaptchainput, checkConnection, gxf, GALX, scc, ProfileInformation, identifiertokenaudio,

(a) Phishing site of a unique variable name vector, indicating the use of a distinct phishing kit or evolution of a kit.



(b) Visually different phishing site, but with a variable vector that's a subset of the vector presented in Figure 2.4a.

Figure 2.4: The evolution of a phishing kit or a phisher's technique.

pstMsg, osid, Page, identifiertoken, profileinformation, ss, utf8, identifiertoken, rm, ltmpl, identifiertokenaudio, historystate0, service, rmShown>

Which fully includes the vector:

<checkedDomains, checkConnection, pstMsg>

Here, either phisher A has learned from phisher B, like built a phishing kit on top of an existing one, or phisher A has improved the attack over time - a claim that's further backed when two seemingly different pages use the same label, the same variable names and target the same brand, like in this example from Figure 2.4.

2.4.3 Case Study 3: Linking Variable Names

Variable names link to individual Threat Actors

Our method leverages the nature of hacker culture where obscure techniques are executed by individuals who hide their traces but still long for credit and reputation. This psychological aspect is not the least reason why our method works and this can be further understood and emphasized by the hacker's banner we found inside an HTML comment: Figure 2.5 shows the same nickname (or handle) from the banner was used to declare variable names and our analysis from Section 2.1.2 successfully detected it as one of the 158 unique fingerprints we exfiltrated from the dataset.

Variable names link to Phishing Kits

Phishers use kits to automate their business. An investigation of existing phishing kits can be challenging because they are not commonly shared in public. While some tools can be found on GitHub we expect the darknet market for it to be much larger and much more sophisticated. However, one kit stands out for being distributed with Kali Linux and its wide adoption by many professionals such as penetration testers. The Social Engineering Toolkit (SET) is arguably one of the most popular phishing kits out there - if not the most - and thus we decided to employ it as a reference point for our case study [62]. SET offers an function to create a template website that mimics Google's *Workspace* login page. That *Workspace* clone is visually similar to the original,

but a look at its page source shows how SET implicitly marks its sites with a unique choice of input field IDs. Figure 2.6 shows the benign landing page, and its SET clone side by side.

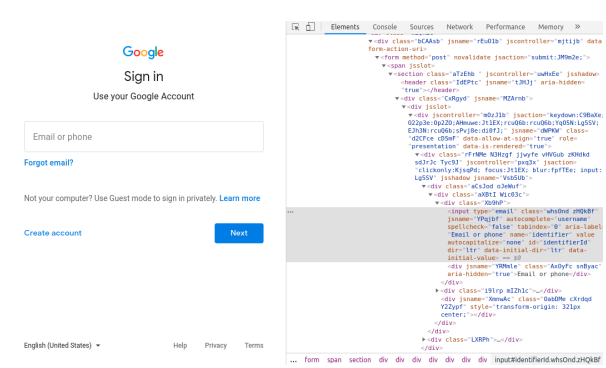
Variable names link to Victim Brands

Other programs are created with less malicious intentions. HTTrack, for instance, does not provide capabilities to administer a campaign or even to harvest inputs. Instead, it solely focuses on generating an exact clone of its input [63]. Motivated by HTTrack we investigate and find that phishers frequently adopt variable names used in the original target page. Figure 2.7 shows this by comparison of an original page, a fake of that brand, and their respective source codes side by side. Specifically, Microsoft uses the variable i0118 for its password fields.

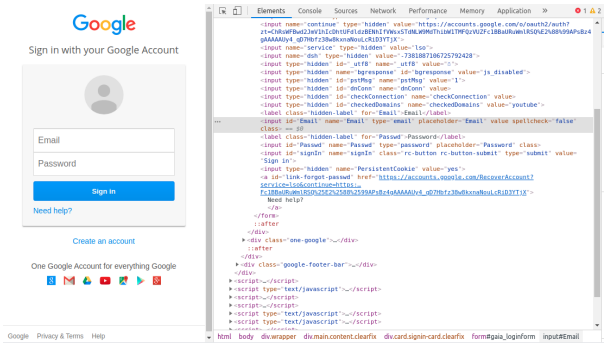
(a) The same hacker's banner as discussed by threat intelligence on Twitter.

(b) A hacker's banner we found by following the variable name analysis.
The hacker was active in 2017 and 2019.

Figure 2.5: A cyber criminal's unique fingerprint.

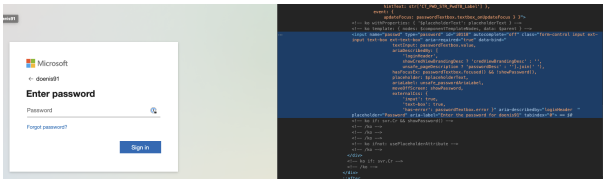


(a) Original Login page at google.com

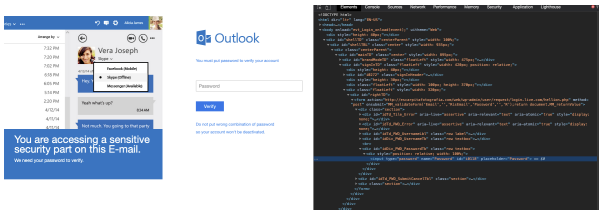


(b) Fake Google Login page created by SET. Using different variable names than the original

Figure 2.6: A benign site and its malicious clone, implementing different input IDs - an artifact of the underlying phishing kit called SET



(a) Original Login page at microsoft.com



(b) Fake Microsoft Login page created using identical (unique) variable names

Figure 2.7: A benign site and its malicious clone, utilizing identical input IDs - an indicator that can be used to link a site to its target brand.

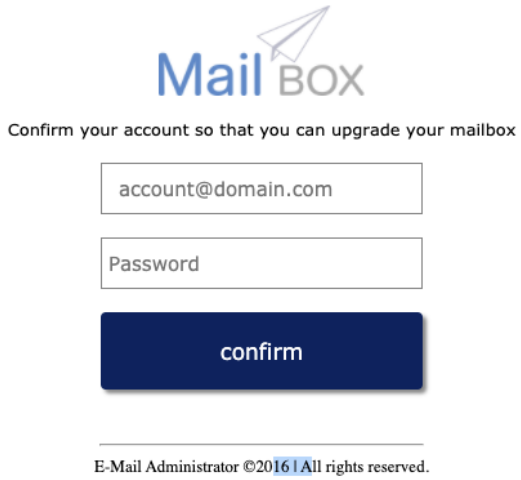
2.4.4 Case Study 4: Variable Names make good clusters

Not all phishing is based on cloning real web pages. Some phishers compose web pages that look like unique landing pages, unassociated with any target brand (see Figure 2.8). Here, the linkage is not possible via visual similarity, but these instances are detectable by the technique of analyzing the input field's ID attribute, which are identical for both subfigures.

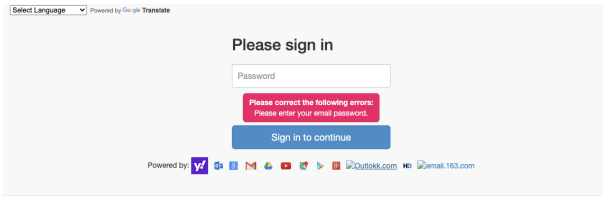
2.4.5 Case Study 5: PII Phishing sites feel more authentic than Credential Stealing

As we will show later, PII Phishing is an increasing trend (see Chapter 2.4.5 Section 3.2. When it comes to the technical implementation of this threat it is striking that even sites with overlapping vectors appear vastly different, with the only commonality being variable names that indicate PII Phishing such as asking for credit card security codes or social security numbers. From our manual

review, we conclude that these websites are much more structured, more authentic-seeming, and more believable. We reason that PII Phishing is more likely to be custom-made as opposed to utilizing a kit. This conclusion is further backed by our observation that those phishinng sites that implement unique input field ID attributes, ask for PII more frequently than sites that overlap with others a lot. Four examples of such professional seeming PII Phishing sites can be compared in Appendix D.



(a) Generic login page.



(b) Different looking generic login page from the same cluster.

Figure 2.8: Two visually different instances of the same phishing kit creating generic webmail login pages.

Chapter 3: Detection, Triage, and Attribution

In the previous chapter, we have learned about two website-features that every phishing site implements. To ask user's for their sensitive data, phishing sites need input fields and these input fields' ID attributes and their descriptive labels make for an interesting characteristic. The previous chapter also motivated the use of these two features for the Detection, Triage, and Attribution of PII Phishing, which we will evaluate in more detail now. Section 3.1 considers the detection of PII phishing using JavaScript stylometry plus the two new features. A thorough evaluation of feature importances quantifies how valuabe the two new features can be for this particular task. In Section 3.2, we use the input field descriptors to categorize which information a phishing site is stealing. The section illustrates a trend study, a language distribution study, and a PII reference list. Lastly, we leverage the two new features as fingerprints of the phishing kit that generated the site, as was motivated in Chapter 1.3.5 Section 2.4.

3.1 Threat Detection

This section will build up from related work to our proposed method of using input field features for phishing detection. We compare the importance of input field features to other JavaScript programming artifacts and we will find that the password fields ID attribute constitutes the most informative indicator of whether a site is phishing or not. Since passwords intuitively comprise the essential piece of information to be stolen in credential theft, this finding confirms our claim that phishers modify the fields they care about and phishers implicitly leave a mark on them, if they are not careful.

3.1.1 Introduction and Related Work

Comparison-based phishing detection has been a popular technique for over 10 years [64] and the similarity between clone (phishing) and cloned site (target) has been studied extensively [27]. Naturally, comparison-based techniques expect the target brand to be known a priori. This work focuses on improving the detection of phishing sites without using brand labels because brand labels introduce bias and do not protect victims who are not represented by a label. We found that related work is based on a rather small feature set of around 30 features and feature engineering has not been discussed recently. Therefore, we investigate a feature set that has not been applied to phishing detection yet. In *De-anonymizing Programmers via Code Stylometry* the authors suggest that programming languages, just like natural languages, are subject to the speaker's individual style [65]. The authors show that a random forest trained on 51 lexical, syntactic and layout features can successfully be used to attribute source code authors based on only a few lines of code. The paper reports 97% accuracy in a closed world experiment, analyzing the labeled submissions of Google Code Jam participants. This method has reportedly been picked up by several government institutions and intelligence agencies and is currently being used to identify threat actors [66]. The method was originally demonstrated for C/C++ and Python code and the adaptation to JavaScript has been proposed but has not been evaluated on real-world web applications yet [67].

3.1.2 Phishing Detection

We are the first, to the best of our knowledge, to apply code stylometry to real-world JavaScript. We reimplement the original algorithm in Python, using scikit-learn's RandomForestClassifier and VarianceThreshold based feature selection. We respect all 51 suggested features, except for word unigrams which have been shown to be secondary and which cause a lot of runtime complexity overhead for large codebases like ours. Our Abstract Syntax Tree (AST) features are based on the esprima project's AST builder for JavaScript [68]. From the analysis of 165,387 phishing sites' embedded and external JavaScript, we learn that code stylometry cannot easily be applied

to de-anonymize phishers. In the closed world binary task, our random forest can only achieve 68% accuracy when deciding if a site is phishing or not. Carefully reviewing the feature selection highlights multiple issues that are unique to real-world JavaScript and do not commonly occur in the originally studied C/C++ code, or artificial codebases:

1. The ratio of proprietary code to common library code
2. White space optimization to reduce network overhead
3. Code obfuscation to hinder detection and unsolicited reuse

While all three characteristics may occur in C/C++ projects we reason that they are more common for JavaScript. Compiled, low-level code like C/C++ is used with performance optimization in mind, which makes it more likely to be handwritten, and even external libraries may be rewritten to avoid linking overhead. On the other hand, white space optimization and source code obfuscation are more of a concern for interpreted code, like JavaScript, where the source code is delivered to the user in plain text. Although we could reason that these characteristics are more prevalent in phishing sites, our results suggest that they are not unique enough to be used as phishing indicators and that they heavily influence the source code style, rendering most of the features suggested by Caliskan et al. redundant.

3.1.3 Evaluation

Figure 3.1 show the feature importance calculated by our random forest classifier. Contradictory to the original findings, AST features are the weakest feature group, as only 3 out of 1,660 possible AST node bi-gram combinations are part of the top 10 most important features. The *average line length* and frequency of some keywords (if, for, new, this, package, function, instanceof) create the rest of the top 10 list. The results are unsurprising since we remember that phishing sites are trying to closely resemble their benign counterparts. On the other hand, we hypothesize that the phisher may adjust the input forms to handle the harvested information. Subsequently, we relax the feature vector to only use the HTML <input> tag's id-attribute. This is promising because

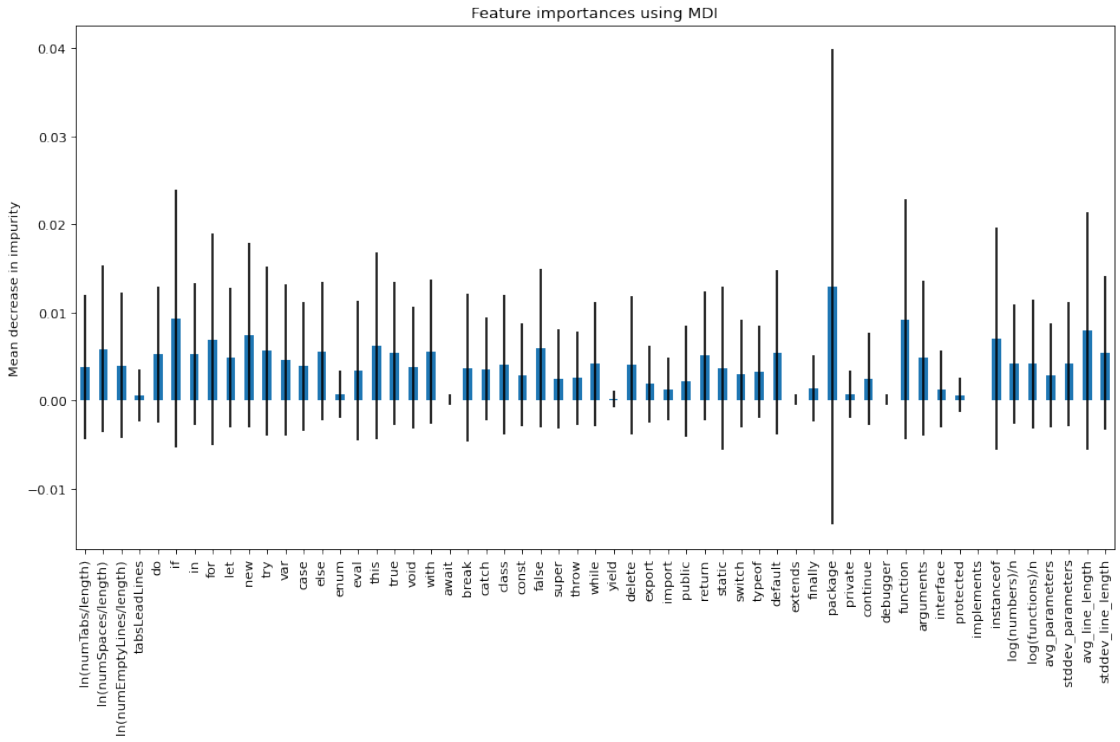
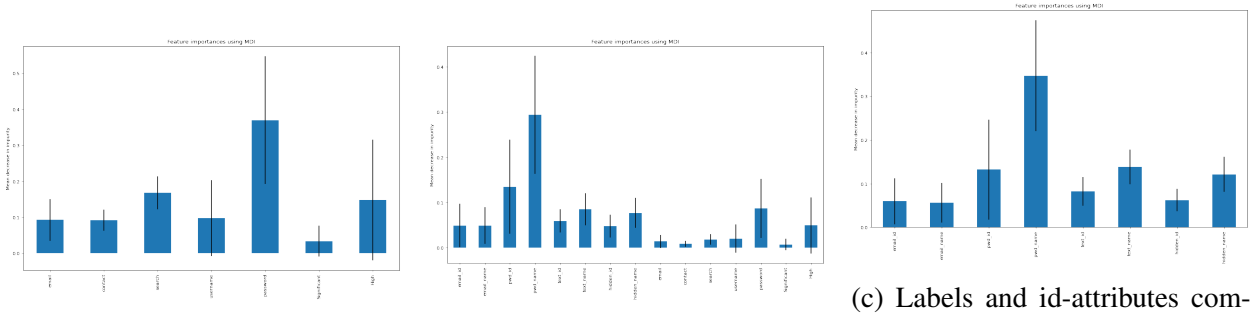


Figure 3.1: Feature importance of lexical and layout features

id-attributes are internally used as JavaScript variable names to store and process the user input. Phishers who operate at scale are likely to use phishing kits to make these changes and integrate the frontend (phishing site) with the backend (verification and retailing). A case study in Chapter 1.3.5 Section 2.4 shows that the phishing kits are likely to leave a mark on the input fields they work with, and this analysis shows that input field IDs and input field labels can be used to detect

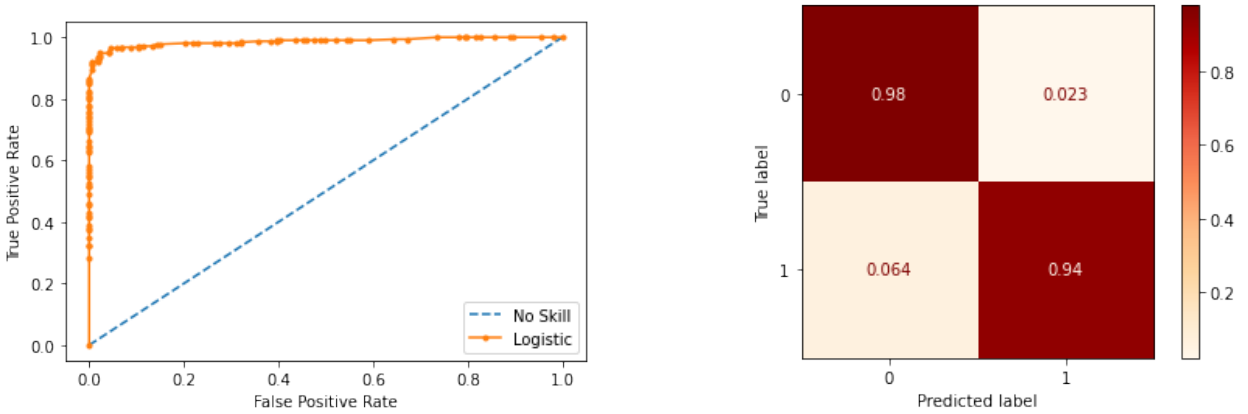


(a) Input field labels only

(b) Input field id-attributes only

(c) Labels and id-attributes combined

Figure 3.2: JavaScript Stylometry features and their respective importance measured. Our two new features (identifiers and descriptors) add significant value.



(a) ROC curve of phishing classifier

(b) Confusion Matrix of phishing classifier

Figure 3.3: ROC curve and Confusion Matrix measuring the quality of our model. It is similarly good at both tasks, making negative predictions correctly and making positive predictions correctly.

a phishing site with 92% accuracy. In addition to the high accuracy itself, we can show that the accuracy was achieved due to the features we newly introduced. The most important features indicate that the labels of password fields and the variables used to store phished passwords contribute to the two most important features (see Figure 3.1). This shows that input field identifiers and input field descriptors can improve over state-of-the-art stylometry features and they add significant value.

3.1.4 Cloaking Detection

The newly introduced feature set has been shown to successfully aid phishing detection in the last chapter. A recent development in phishing is the use of detection evasion techniques, so-called cloaking. This chapter considers the use of HTML <input> tag id-attributes for the classification of phishing sites as cloaked or not, and it shows that the feature may help to decide if and which cloaking technique a phishing site is using. This experiment bolsters the hypothesis that HTML <input> tag id-attributes expose information about the phisher's technology stack.

Phishers naturally aim to avoid detection. Therefore, modern campaigns analyze their visitors to decide which content to present. This strategy is commonly referred to as cloaking and has been investigated extensively in recent work [70], [71]. From a high-level view, it has to be dif-

ferentiated between server-side techniques to reject certain connections [30,37,44] and client-side cloaking techniques that use JavaScript to profile the victims behavior on the site. The popularity of JavaScript for client-side cloaking makes it a perfect use-case for our newly introduced method, the evaluation of a sites JavaScript Sylometry. Client-side cloaking can be particularly dangerous because it enables the implementation of complex interactions with potential victims.

CrawlPhish: Large-scale Analysis of Client-side Cloaking Techniques in Phishing by Zhang et. al. defines a set of eight different client-side cloaking techniques, namely, User-Agent, HTTP referrer, time zone, geolocation, mouse movement, cookies, random access, captcha, and popup all are used in the wild. Note that this list also supports our reasoning about offensive deception techniques in Chapter 6, saying that phishers verify their victims and stuffing engines will have to overcome some obstacles.

Methodology

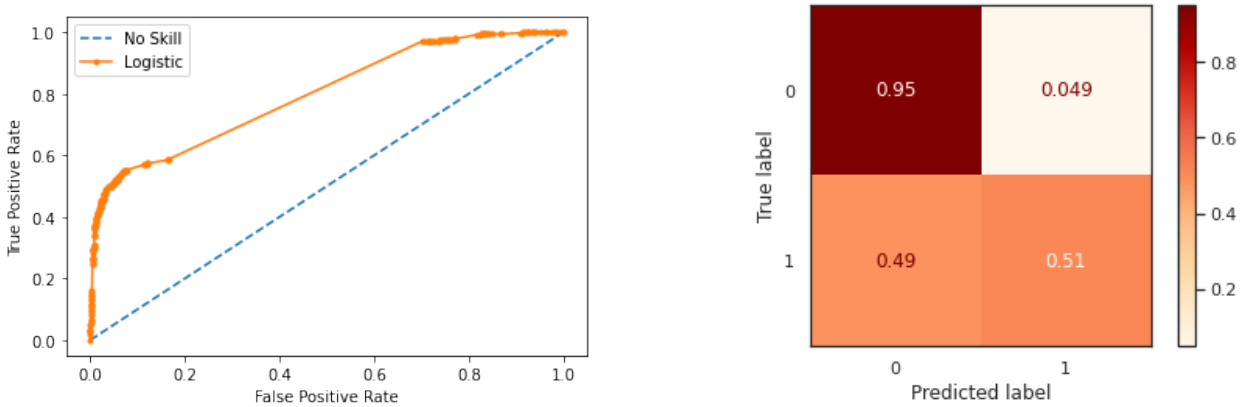
To evaluate the use of JavaScript Stylometry features for cloaking detection, we propose to reuse the dataset collected by the CrawlPhish project. The dataset is compiled of 42,123 HAR files and a list of the features and labels that the authors used for their own research. We will extract the HTML and JavaScript from the HAR files, such that we can parse the JavaScript for stylometry and the HTML ofr input field identifiers and input field descriptors. Using this feature set, which we originally introduced in Chapter 1.3.5, we will then explore the following experiments:

1. Binary Cloaking task
    - CodeStylometry only
    - CodeStylometry + labels + identifiers
2. Muli-Class Cloaking task
    - CodeStylometry only
    - CodeStylometry + labels + identifiers

A random forest classifier is trained on each of these feature sets. From the original dataset, we select such instances that either use JavaScript through external linking, or embedded in the HTML via <script> tag, or both. Of those source code files we keep the one that can be successfully processed by our AST parser, which detect unknown node types in 5.8% of them. In the end we work with 13,397 sites that use external scripts and xxx sites that embed the code. Note that some sites use multiples <script> tags so we work with 249,266 script and count the features across those that came from the same HTML.

Like in the last chapter, we extract 1600 AST node bi-grams, 54 lexical and 6 layout features. Then, we reduce the feature set by removing all features that do not meet a variance threshold. The threshold is set to cut off features that are 0 or 1 more than 80% of the time and this threshold reduces our feature set from 1661 to 400. Then, we split the data into 80% training and 20% test cases. and use the training dataset to train a random forest model with a maximum tree depth of 9 nodes, at each node, the tree considers up to $log_2(\#features) = 9$ features, evaluating each possible split's information gain using Shannon entropy; and using out-of-bag samples to estimate the generalization score.

In a second experiment, we split the binary labels into the specific cloaking techniques that the phishers used. We use the same classifier configuration, but now we predict 71 labels instead of



(a) ROC curve of cloaking classifier.

(b) Confusion Matrix of cloaking classifier.

Figure 3.4: ROC curve and Confusion Matrix measuring the quality of our cloaking detection model. It is particularly good at predicting *un*cloaked sites as *un*cloaked.

Class	0	4	6	7	8	9	10	11	12	13	14	15	16	18	19	20	25	26	27	28	29
Precision	0.73	0.82	0	1	0	1	0	0	0.6	0	1	0.75	0.7	0.45	0.85	0	0.5	0	0	0	0.75
Recall	1	0.07	0	0.5	0	0.67	0	0	0.28	0	1	0.6	0.88	0.97	0.79	0	1	0	0	0	1
F1 Score	0.84	0.13	0	0.67	0	0.8	0	0	0.38	0	1	0.67	0.78	0.62	0.81	0	0.67	0	0	0	0.86

Class	30	34	36	37	38	39	42	43	44	50	51	53	56	57	59	62	63	65	66	69	70
Precision	0.83	0	0	0	1	0.68	0	0.93	0	0	0.25	0.57	0.54	1	1	0.67	0.5	0	0	0	1
Recall	0.99	0	0	0	0.05	0.42	0	0.7	0	0	1	0.48	0.3	1	1	0.67	0.33	0	0	0	0.25
F1 Score	0.91	0	0	0	0.1	0.52	0	0.8	0	0	0.4	0.52	0.39	1	1	0.67	0.4	0	0	0	0.4

Accuracy	0.52

Table 3.1: Model metrics for the cloaking detection binary class

two. All 71 labels are found in the original *CrawhlPhish* Dataset and each of them represents either one of eight cloaking techniques, or a combination of multiple cloaking techniques. For the multi-class tasks, the random forest achieves an average accuracy score of 52%. Table 3.1 summarizes each quality metric per class.

3.1.5 Discussion

We will now discuss the implications of our measurements. For the binary tasks, the confusion matrix in Figure 3.4b suggests that the model is very accurate at confirming if a site is *not* cloaked, which it tells with 95% confidence. For cloaked sites however, the model performs almost as bad as guessing. We see that the model classifies a cloaked sites as cloaked or not with a ratio of 49 to 51. The ROC curve in Figure 3.4a shows a steep increase, meaning that the classifier makes confident decisions up to about 58% accuracy before it starts to make more mistakes. From there, the curve makes a jump, meaning that it can only make more good predictions at a very high risk of making false predictions. Knowing the strengths and weaknesses of this model, we envision to use it as a preprocessor for an un-cloaking system. When threat analysts want to get past cloaking, many different methods have to be tested, and significant time effort has to be made, not the last expensive manual reviews. Knowing that a site is unlikely to be cloaked can tell the analyst not spend too further resources on them..

3.2 Threat Triage

In the last section, we have seen that the two newly introduced features, input field IDs and input field descriptors are valuable indicators of whether a website is phishing or not. It appears that the phishers use of phishing kits that predeclare these values for post processing purposes and better integration with the rest of the kit. Especially the ID attribute associated with password fields tells if a site is a threat. In this section we investigate the other one of the two features, the input field descriptor. The descriptor is a plaintext user-facing text value to tell a website visitor what information to enter. A collection of all input field descriptors summarizes the malicious site's implicit goal, and it thereby implicitly leaks information about the phisher, like e.g., which information the phisher is stealing may tell which sort of crime the phisher engages in. We propose to use the new feature for the triage phishing sites into different threat categories. Knowing the purpose of an attack provides an additional data point to prioritize take down efforts and to potentially notify impacted parties what to be wary of. We will now introduce how phishing is mitigated today, showing that a strong focus on password security yielded a shift from Credential Theft to PII Phishing in Section 3.2.2. Another insight of the descriptor analysis can be which nationalities the phisher target (see Section 3.2.4) and we summarize all or findings in Section 3.2.5 to build a PII reference list that can be use to implement data protection rules.

3.2.1 Introduction

Promising anti-phishing solutions, such as Multi-Factor Authentication (MFA), the FIDO standard, public blocklists, email, and call filters are evidence of a recent race to mitigate credential theft - not just for corporate network users, but also for public Internet users. At the same time, these solutions may convey a false sense of protection and distract from other threats. As a commonality, all phishing attacks exploit the human factor but their ultimate goals can be very different. While some phishers are looking to make a quick buck from careless end users, others aim to infiltrate foreign governments, perform industry espionage or run underground crime by setting up

mule accounts for money laundering, drug deals, human trafficking, or terrorism. Stealing account credentials is only one way for criminals to achieve these objectives. Stealing personally identifiable information (PII) is an overlooked but ongoing and ever-growing threat that allows criminals to impersonate their victims. It can pose tremendous financial consequences to the impersonated victims including the cost and time it takes to initiate or defend against legal actions, the potential impact on credit scores, and the unrecoverable loss of reputation. Recently we see government and law enforcement agencies collaborating to prosecute e-criminal activities [], but their job is difficult since many criminal activities occur within unfriendly national borders and are geo-fenced to avoid attacking their hosting country. Even server take-down requests can be vain if the hosting organization does not collaborate because the investigator does not provide the right evidence. In a race to prevent crime, and protect innocent victims, it is important to gather extensive threat intelligence and not leave any trace unnoticed. The academic community has intensely studied whether a website is a phishing website or not and we propose to extend this question by asking: What is being phished for? In this paper, we analyze the content of a confirmed phishing site and represent it as a word vector of form field labels that indicate what information the offending site asks for.

3.2.2 Triage Schema

We triage phishing landing pages based on the observation that phishing has matured beyond credential stealing and is increasingly being used for identity theft. Typically, phishing campaigns may either target the general public, or carefully chosen individuals (spear-phishing/ whale phishing). Regardless of the victim, every phishing campaign results in the victim's loss of sensitive information. In this study, we do not analyze which victim class experiences a higher threat or which of these campaigns have a higher impact. Instead, we focus on whether the stolen information can be used to impersonate the victim, namely, we triage based on what information is targeted. Understanding the risk of identity theft is a valuable first step toward detecting and mitigating its threat. The following rating scheme without loss of generality delineates phishing websites according to the sensitivity of the information they steal:

Contact Gathering Gathering of (publicly available) contact information for subsequent targeted attacks, but without tricking the user into providing secret/private information. For example, phishing websites that pose as coupon sites may gather email addresses, without passwords, for use by the phishers in later campaigns, or for sale on the black market for future phishing campaigns by other phishers. Similarly, phishing by invitation requests only the user's mobile number for future targeted vishing campaigns.

Credential Stealing Gathering of user credentials that provide access to some valuable third-party resource, account, or service. Here, the phishing website may convince the user to provide their login account and password which provides access to their banking website, and hence, access to their bank account. Notice that no other PII information is requested by the site.

Identity Theft A phishing website may be designed to gather the victim's sensitive PII, sufficient to steal the victim's identity and create new accounts using that stolen identity. Identity loss can impose significant financial losses and years of effort to deal with repairing the damage. Some phishing websites that are spoofed, or cloned from a legitimate source, request sensitive PII well beyond what the original legitimate site required. For example, a user's banking website is not likely to require their complete social security number each time the user logs on.

Intuitively, the three categories constitute a hierarchy of threat levels. Contact gathering collects publicly available information, likely resulting in either spam or follow-up phishing attacks. Credential Theft is about platform specific access tokens and only some cases may lead to a higher threat of identity theft if PII is stolen from within that account. PII Phishing poses the highest threat based on the following three observations:

- PII Phishing escalates the threat from the online world to the physical world (as opposed to credential theft)
- PII Phishing can be used for identity theft which may go unnoticed by the victim (as opposed to scam purchases)
- Stolen PII can be used repetitively and in multiple locations (as opposed to one-time scams)

We implement this three tier model and assign each website from our dataset (see Section 2.2) to its exclusive threat level. Each threat level is defined by a subset of 36 information categories, that we manually derived from the dataset. Each information category is indicated by one or more of 356 unique category indicators that we use to summarize the actual terms extracted from the website. For example, a site that asks for the indicators "e-mail" or "email", is mapped to the information category *email*, which is evidence of contact gathering (see Figure 3.5 ❶&❷), or credential theft, if asked in combination with a password (❸&❹).

While we are aware that our category list cannot be complete, we have met a level of granularity that allows us to attribute each of the 66,634 labels (8003 of which were unique) to at least one category. We achieved this by starting with a brainstormed reference list of PII categories, mapping HTML labels to those categories, and manually reviewing unmapped labels, iteratively adding their respective categories and indicators to the list. To the best of our knowledge, there is no equally detailed list of PII categories available today.

The hierarchical treemap in Figure 3.5 shows a series of nested rectangles of sizes proportional to the corresponding frequency of label terms in our dataset. The outmost large rectangles illustrate information categories, framing several indicators, illustrated by the smaller rectangles. The mapping from indicator to category is further improved by a set of rules to handle edge cases. For instance, we ensure that any name indicators in fact indicate legal names, not usernames, and address indicators in fact indicated the physical mailing address, not the email address. Some other indicators like ”Last 4-digits of . . . ” are also ambiguous. E.g. they can refer to either an SSN or a bank account. However, we have found that all seemingly ambiguous indicators map to categories that pose the same type of risk. Therefore it would not skew the threat category triage. In other words, they form tolerable false positives at worst. Noticing that our best-effort approach is prone to errors, we invite the reader to manually review the quality of our analysis. A complete triage of our dataset is hosted online. The interactive sunburst diagram shows each phishing site’s label word vector, the PII category mapping, and its threat level. In the future, we will add a feature to click an instance to open the original phishing site, too.

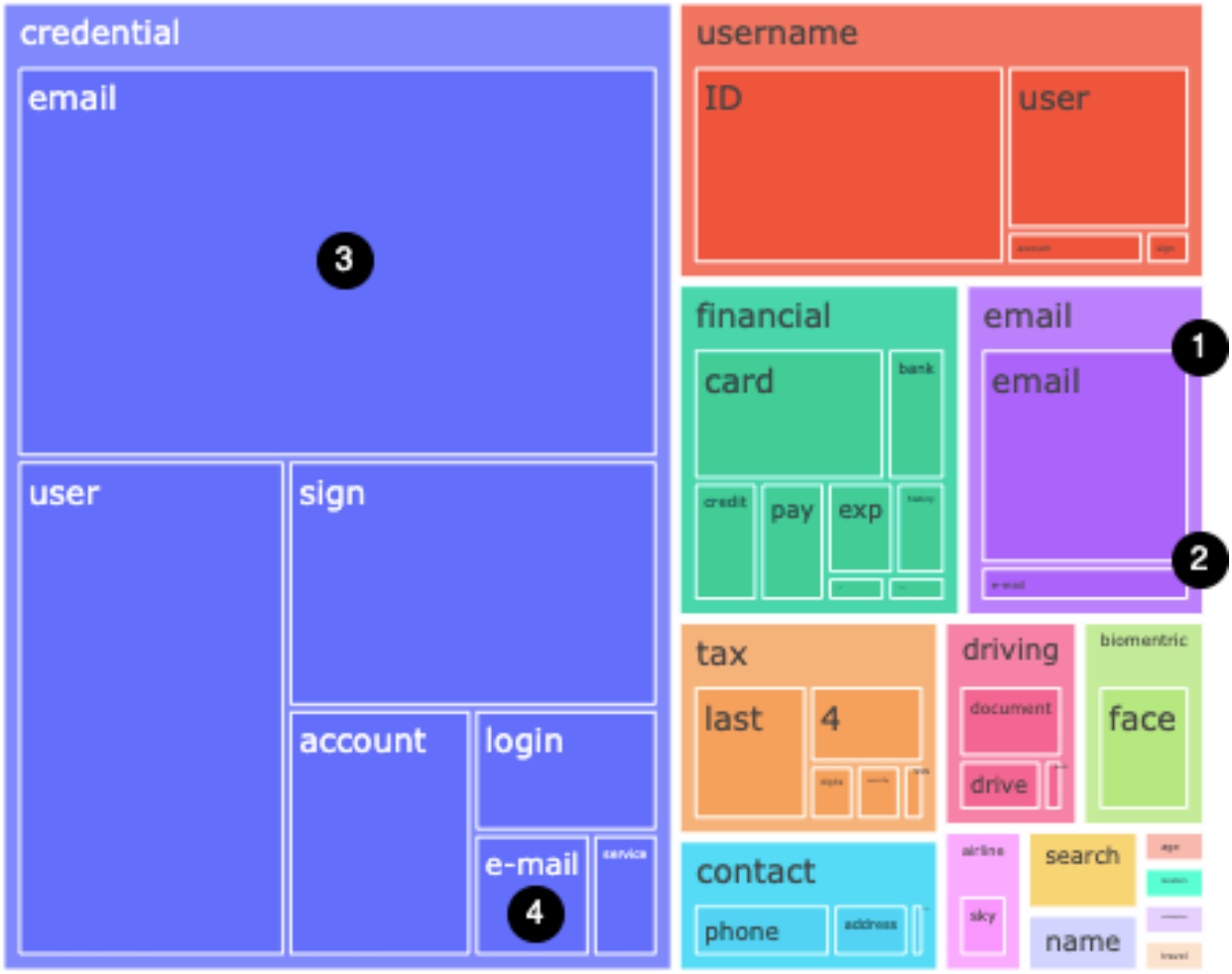


Figure 3.5: Overview of 36 information categories. each of which contains multiple indicatorss.

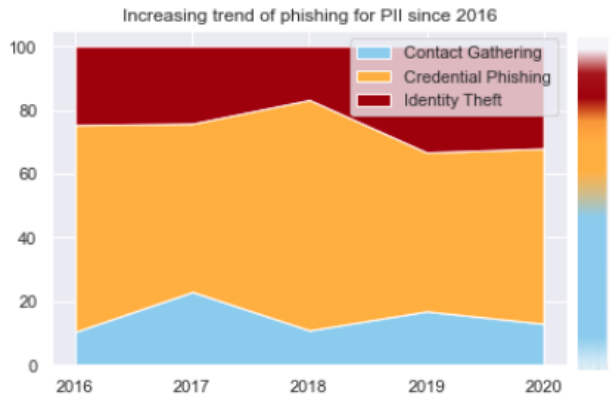


Figure 3.6: Five year trend showing what information the phishing sites in our dataset target over time.

3.2.3 The Trend of PII Phishing

We triage all sites in our dataset as one of the three threats. which enables a longitudinal trend study showing how threat levels evolve over time. In Figure 3.6, we have measured the frequency of each category over five consecutive years.

The graph shows that traditional Credential Theft is on the decline and it is gradually being replaced by PII Phishing. In comparison, it is very rare for benign websites to inquire about this kind of information. The right plot shows a threat level distribution among our ground truth data set of benign sites. Only less than 20% of benign sites ask for PII while asking for low-risk information is much more common there (>60%). Credential Theft (center) declined from 72% to 54%. Identity Theft (top) increased from 16% to now 31%. Contact Gathering (bottom), remained at around 15%.A baseline dataset recorded in the year 2017 shows how common Contact Gaterhing (bottom) is among legitimate/benign websites and how rarely they ask for PII (top).

3.2.4 Natural Language Distribution

In addition to the categorization of keywords, the victim-facing input labels reveal another characteristic: the natural human language used by the campaigns. Language is an unambiguous indicator of the phishing campaigns' target audience. We used Python langdetect module [69] to

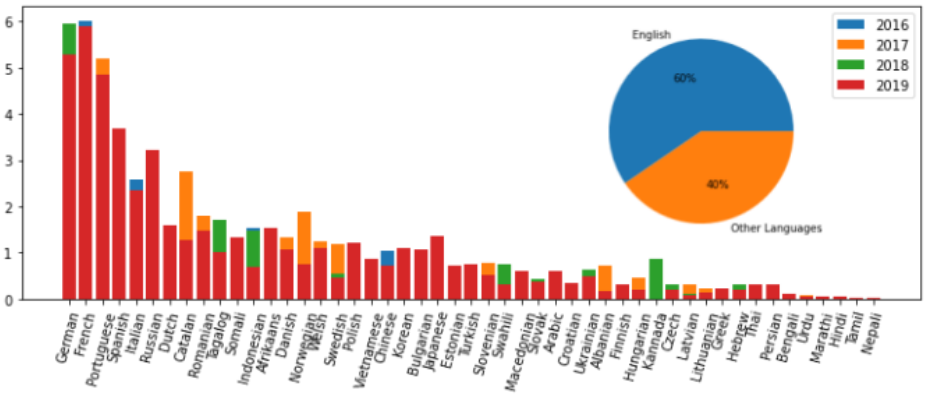


Figure 3.7: The datasets distribution of natural human languages.

detect languages from input field descriptors. The majority of sites are in English as the pie chart in Figure 3.7 shows. Other languages are plotted as a bar chart using different colors per year. We found that over half of the campaigns are crafted in English. We also find 48 other languages including widespread languages like German which has over 100 million native speakers in the DACH region and French, which is not only spoken in France but also in the Arabic World and Francophone Africa. We also see a high representation of Russian and Portuguese which can be explained by high levels of cybercriminal activity in Russia and Brazil [70, 71]. The report Usage statistics of content languages for websites [72] allows us to compare the distribution of phishing languages to ground truth distribution of languages on the internet (see Appendix Table C). These statistics highlight that phishing is a global problem affecting all languages and stealing all types of information.

3.2.5 PII Reference List

There is general agreement on what constitutes sensitive personally identifiable information, but the abstract idea discussed by law and policy-makers does not provide actionable reference lists [73, 74, 75, 76]. To implement data protection in practice, such a list is crucial, so many organizations maintain their own [77]. Some policymakers distinguish between private and public PII, where ”public” includes, e.g., birth date and address and ”private” includes financial and

medical information, among others. This distinction depends on the context. Information such as employment status can be public, or not depending on how much a person shares on social media, like LinkedIn. By reversing what the phishing sites in our dataset attempt to steal we derive an initial list of identifiers. We then iteratively refine our list by reviewing all sites not yet covered by our identifiers. We propose a granular categorization of 36 categories derived from (currently) 356 unique identifiers and we compare this categorization to existing lists, like NIST [78] and Google Cloud DLP [77]. Google Cloud DLP provides a list of 132 categories to de-identify data, however, 93 of these categories are country-specific identifiers for tax, medical and government IDs, so in our notion, they constitute indicators rather than categories. Others are technical identifiers or machine identifiers, like IP address, or authentication tokens, like web cookies. In our notion, Google Cloud DLP presents only 28 PII categories that we propose to extend with 11 additional categories of our own:

1. Education
2. Origin
3. Genetic
4. Biometric
5. Family Status
6. Employment
7. Preference
8. Brokerage
9. Travel
10. Airline
11. Toll

The categories AIRLINE and TOLL are service-specific categories, so their sensitivity depends on the context, but other categories we add are clearly worth protecting. For example, TRAVEL information includes redress numbers that are used to enter countries, PREFERENCE includes sexual orientation that may lead to persecution in many parts of the world, additionally BIOMETRIC, GENETIC and information of ORIGIN that can be used for impersonation. Our comprehensive PII reference list appears in Appendix A where we directly compare it against the Google Cloud DLP list.

3.2.6 Discussion

PII Sensitivity

To rate the sensitivity of PII, one may assign different instances of PII, or combinations of PII, to a different threat level as the context may require. Each phishing website analyzed would be assigned the appropriate threat level based on this assignment. Sensitive personally identifiable information includes:

- Employee personnel records and tax information, including Social Security number and Employer
- Identification Number
- Passport and government ID information
- Medical records covered by HIPAA laws
- Credit and debit card numbers
- Banking accounts
- Bitcoin/Wallet Accounts
- Electronic and digital account information, including email addresses and internet account numbers

- Passwords
- Biometric information
- School identification numbers and records

Hence, if a phishing website analysis reveals that some of these digital identity attributes are being gathered, that site is considered extremely dangerous since sufficient information is available to create a new digital identity. Other phishing websites may be deemed highly dangerous but not extreme if only non-sensitive PII is gathered. Non-sensitive PII is generally publicly available information and includes:

- Birth dates
- Place of birth
- Addresses
- Religion
- Ethnicity
- Sexual orientation
- Business and public personal phone numbers
- Employment-related information

Other phishing websites may be deemed to be worrisome, even if they do not target sensitive PII nor non-sensitive PII. For example, a single email address without a password is troublesome, but ultimately it may only pose a threat to the user's spam folder.

Relevance for Law and Governance

Industry regulations such as GDPR, CCPA, CPRA, and HIPAA approach the concern of consumer data protection through demand for compliance[79, 74, 80, 73]. Additionally, industry-specific regulations address edge cases of PII usage in trading, marketing, consumer credit evaluation, and many others [81, 82, 83, 84, 85, 75]. To the best of our knowledge, these acts employ individual, normative definitions of PII by indicating similar intent. One anchor-definitions that's frequently used as a reference can be found in The Code for Federal Regulations (CFR 2 § 200.79 [76]), which normatively defines PII as follows: PII means information that can be used to distinguish or trace an individual's identity, either alone or when combined with other personal or identifying information that is linked or linkable to a specific individual The CFR definition also distinguishes PII that's publicly available, it continues:

> *Some information that is considered to be PII is available in public [...] includes, for example, first and last name, address, work telephone number, email address, home telephone number, and general educational credentials.*

However, the definition emphasized that the categorization of information as PII or

> *Non-PII has to be met dynamically, as circumstances may change: Non-PII can become PII whenever additional information is made publicly available, in any medium and from any source, that, when combined with other available information, could be used to identify an individual.*

And therefore it concludes:

> *[...] it requires a case-by-case assessment of the specific risk that an individual can be identified.*

The normative, non-descriptive, regulations that define PII refrain from concrete suggestions on how to assess PII in practice. Hence, we refer to The National Institute for Standards in Technology (NIST) as an (independent) reference guideline for PII assessment and PII protection. The NIST

suggests assessing PII based on four external factors: Quantity, Quantity of Harm, Context of Use, Access to, and Location of PII. Once assessed the NIST suggests concrete methods to protect the sensitive information:

- Purging unnecessary PII from records.
- Implement access control measures.
- Encrypt all sensitive information.
- De-identifying (anonymizing) data and feedback so that it cannot identify.

While the first three methods can be statically applied once a piece of information is categorized as PII the de-identification of automatically processed data at scale is a big- ger challenge. Cloud providers such as Google and IBM [77] have picked up on the issue and offer products to automatically detect and de-identify PII without data loss [86]. For automated detection, the service providers have to leave the abstract domain of norms and regulations and convert the descriptions into actionable implementations. For example, Google Cloud DLP employs more than 132 distinct classifiers to filter sensitive information in customer data [87]. Accordingly, state-of-the-art actualizations of legal compliance are achieved through a finite list of identifiers. When answering the "What?" part of our research question, we find 13 additional information classes that fraudsters have an interest in stealing, which means that the information is valuable for impersonation and which thus can be considered PII. Subsequently, we propose the following alternative definition for PII: "Any information that is valuable to and targeted by Identity Theft constitutes PII."

3.2.7 Summary

Phishing is typically considered a credential-stealing attack. That is not correct. The overwhelming majority of phishing sites are typically designed to steal digital identities allowing attackers to impersonate victim users, rather than simply gaining access to their accounts or services. An analysis of over 131,023 phishing sites indicates a rising number of over 31% of those sites

pose an extreme danger since they are designed to steal a victim's PII. A declining percentage, of 54% on average is primarily designed to steal credentials posing a high danger to users. A far smaller percentage of 15% trick users into providing contact information alone, such as a phone number or email address, likely for future use in phishing, smishing, and vishing campaigns. The ease of using automated phishing kits has made phishing a global phenomenon with fraudulent sites deployed in 49 languages. English still dominates the collection of phishing sites, with a substantial number of substantiates in recent years growing in languages such as French, German, and Portuguese. The cautionary tale for companies is that their websites are being used by phishers to steal their customers' identities, not just to gain access to their customers' accounts. Two-factor authentication and FIDO do nothing to stop customer identity theft. Consequently, their brands may be at risk of inheriting liability for identity theft. A key contribution of this work is the automatic acquisition of ground truth data about malicious attackers based on their own programming style and tools used in large collections of phishing sites, and the automatic evaluation of the threat level a phishing site poses based on which kind of PII the site attempts to steal from its unwitting victims. The evaluation metrics have been outlined in general terms of three distinct levels of danger, but may be extended to finer granularity depending upon the context. Furthermore, profiling attacker behaviors has tremendous value as advanced threat intel for defenders seeking fast detection of likely adversary threats. We developed detailed profiles of the code within each phishing website to identify clusters of different websites likely created by the same phisher, or phishing team. The analysis is greatly simplified by focusing entirely on input variable names and terms displayed to the user. This provides insight into the number of distinct phishers in the dataset, and provides data for longitudinal studies, as well as data for predictive analysis of what phishers do over time. The method of extracting phisher-defined variable names may also be an excellent indicator to quickly detect likely phishing sites.

3.3 Threat Attribution

We have now seen the use of input field IDs and input field descriptors for the detection and the triage of PII phishing sites. Our case study in Chapter 1.3.5 Section 2.4 also motivated the attribution of a phishing site to the phishing kit it was created by. We will further investigate this use case in this following section. In particular, the examine relationship between phishing sites that may have been created by the same phishing kits or by related phishing kits.

3.3.1 Introduction

We have to understand an enemy in order to defeat him, prominently suggested by Sun Tzu in The Art of War [88]. Enhanced Attribution applies this philosophy by analyzing unique features of a threat and linking them to the threat actor behind it, a valuable step towards tackling PII Phishing since the severity of phishing lies in its uncertain nature. Not knowing what information was stolen and whom it is used by makes it much harder to mitigate the impacts that may only become apparent years later when the original incident is long forgotten. Personal circumstances and soft values like credibility or reputation may be costly or even impossible to restore. As identity theft is a matter of national security, active take-down programs and enhanced attribution campaigns are also run by government agencies [66]. However, there is no global unity on how to solve the issue and in some countries, phishing may not be illegal or is even encouraged [89, 90, 91], other countries may simply not have the infrastructure or law enforcement capacities to prosecute phishers. Allowing for organized crime and unashamed fraud operations. Although it is often believed that all online fraud originates from countries with the aforementioned shortcomings, news reports show that many large scale threat actors get caught in Western Europe and the United States of America, but they may hide parts of their operations or host servers on the other end of the world [92, 93, 94]. Uncooperative hosters, VPN and proxy providers as well as ultimately the Tor Network provide sufficient disguise to avoid enhanced attribution. Further, phishing sites are not necessarily self-hosted but squatted on hacked, benign servers [95]. Extremely short life

cycles further complicate the analysis of phishing campaigns [39]. With all legal, IT, and network fingerprinting-based methods being exhausted, we propose to take a closer look at the additional information fraudsters reveal about themselves: The phishing sites source code, including its input field identifiers and input field descriptors as we motivated in Chapter 1.3.5.

3.3.2 Social Graph of Phishing Kits

Like every professional, criminals tend to specialize. Hence, the tactics of threat actors who engage in PII phishing likely vary from those of solely financially motivated scammers. While it is almost impossible to pinpoint individual actors (see Section 6) and categorizations such as script kiddies, government institutions, or terrorists are subject to interpretation and likely to be biased, we focus on what we can objectively quantify. This study is interested in who acts underground, who is connected to whom, and who is learning from one another. We hypothesize that phishers write - and share - phishing kits that are refined and adjusted over time and for different campaigns' needs. We investigate the use of the same victim-facing input field identifier attributes as an artifact of such phishing kits and we find that many of them are in fact connected. As motivated by the feature engineering chapter, chapter 3, we can generally represent a phishing site (or really any website) as a sorted word vector. For the technology fingerprint analysis, we extract identifiers rather than labels. In this highly reduced representation, we maintain the original structure of each vector, when we compare two vectors, however, the order of words becomes less important. We assume that phishers either fully reuse, or don't reuse existing code. Hence, comparing two vectors is modeled using case sensitive (and stripped) equality, and we refrain from any similarity or distance measure between words themselves. They are either fully equal or not equal. For instance, "email", "Email" and "mail" are fully orthogonal and share as little commonality as "usr" and "pwd". Figure 1a shows the frequencies of vectors found in the 2019 dataset and highlights in bold letters how vectors 1, 2, 5, and 8 are connected. The undirected weighted graph in Figure 1 shows an edge between nodes that have one or more elements in common. The number of shared elements is used as edge weight signifying the strength of an overlap. We hypothesize that we can

use the overlap of identifiers to map out a social network of phishers who are sharing their kits, learning from each other, and improving themselves.

Unfortunately, the data for this social network is dirty, and it is not intuitively clear which node has more influence. For example, we can analyze a social network from several perspectives, such as which node has the highest outgoing edges (degree), participates in the most complete subgraphs (cliques), or connects the most other vertices via shortest paths (betweenness). To the best of our knowledge, the CorpRank algorithm [96] is the first algorithm to consider all (!) of the above-mentioned features. CorpRank leverages Principal Component Analysis (PCA) to derive additional weight coefficients for the features.

$$CorpRank(node) = \frac{\sum_{i=1}^{m} w_i X_i}{\sum_{i=1}^{m} w_i}$$

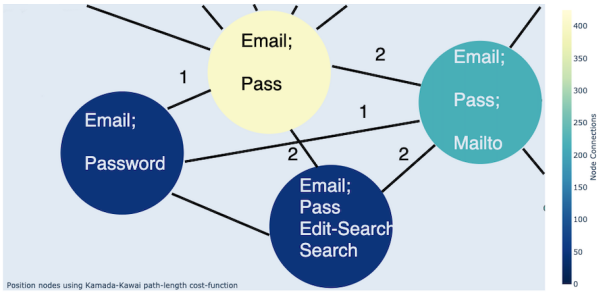
$$w_i = \frac{\sum_{j=1}^{n} \beta_i^j}{\sum_{i=1}^{m} \sum_{j=1}^{n} \beta_i^j}$$

$$PC^j = \beta_1^j X_1 + \beta_2^j X_2 + ... + \beta_i^j X_i + ... + \beta_n^j X_n$$

The score provides a hierarchy of influence, which in practice, can serve as an additional data

2019
1. **email**; password (1343x)
2. **email**; **pass** (854x)
3. atok; u (830x)
4. xpass; xuser (823x)
5. email; **pass**; mailto (623x)
6. ipAddress (553x)
7. password; username (439x)
8. edit-search; **pass**; **email**; search (416x)
9. usr; psw (398x)
10. s (397x)

(a) An example of the most frequent vectors in 2019 and their respective frequencies.



(b) Graph model showing phishing kits as word-vectors and edge weight based of shared terms. The same concept is used to plot Figure 7.

Figure 3.8: An example of how the second most frequent vector in of 2019 is presented in the network. The connecting edges are weighted based on the shared elements "email" and "pass".

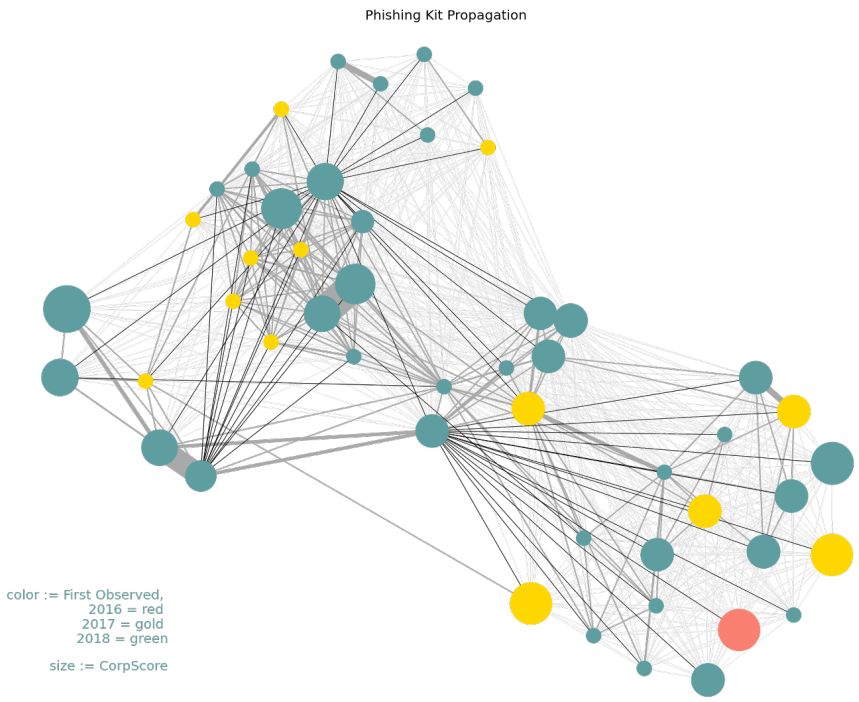


Figure 3.9: Social Graph of Phishers

point for threat actor attribution. It can lead investigations towards related attacks, or guide decision making by revealing whether a site seems to be hosted by a lone wolf indicating a sophisticated targeted attack or a common spray'n'pray strategy - indicated by a low or high CorpRank respectively.

In Figure 3.9 we present how we use CorpRank to add structure to a social network of phishing kits. The undirected weighted graph resizes each node by its CorpRank score and sets the edge weight to the number of input field identifiers shared between two kits. The node's color maps to the year that it was first observed in and we highlight the minimal spanning tree of the graph with black edges. Figure 3.8 zooms in on a node and adds details for a better understanding.

When reviewing the CorpRank score of each node, by looking at its size in the plot we can roughly say if a node is small medium or large and we get an idea for its influentialness. We may reason that a large node, is a popular kit that a small node copied from. In practice, the score can be used as a numeric value and we can use it to guide digital forensics operations. For example, this knowledge can be used whenever a new threat is detected. Linking the new threat to an existing

one is a promising step to gather intelligence about it and the CorpScore-based social graph can help to prioritize which known threats to consider the most.

Chapter 4: Offensive Deception

In the preceding sections, we present how JavaScript Stylometry can be used for the Detection, Triage, and Attribution of PII Phishing Campaigns. We evaluate the importance of a newly introduced feature, the HTML input form identifier attributes, and we show that it can significantly improve the accuracy of a phishing classifier. We also show proof for the hypothesis that identifiers reveal information about the threat actor and we organize this information into a social network of phishing kits that are built upon one another. Without a ground truth set of phishing kits, this social network can only serve as an unsupervised clustering model, so we investigate further to determine the relative importance of each node (phishing kit instance) using the CorpRank algorithm. We also show that the related HTML label tags can be used for the triage of PII phishing and we derive a unique PII reference list. The final chapter of this thesis will focus on a denial-of-phishing engine (DOPE) that submits fake data to phishing sites. DOPE leverages the threat triage system's label analysis and automatically generates decoy information according to a site's expectations. We further establish evaluation sensors to monitor if decoy information was picked up by the phisher and we perform A/B Testing to compare the different attack parameters. This comparison provides insights into the state-of-art of phishing techniques and it serves the evidence for several common industry claims such as that phishers test their catch.

4.1 Introduction

The detection of an attack is crucial, but it does not leverage all that is technically feasible to mitigate a threat and prevent further damage. One popular form of incident response is to deceive the attacker with fake information and divert her focus from the actual target. Deception systems date back to 1998 [97] and have been proposed for all layers of a system (e.g. HoneyPots [98], HoneyTokens [99], bogus credentials, decoy documents [100], fake network traffic [99], and

source code traps [101]). Some projects go as far as hosting purposefully vulnerable web applications to enable phishers to exploit them and host phishing campaigns [102]. This way the threat analysts have studied the complexity of phishing kits and proposed to cluster their interdependence [103], and cloaking scripts [7] the phisher's use of proxies and tor [7]. This method has further discovered that most compromised hosts get abandoned by the phisher after less than 21 hours, which dates the average lifetime of a phishing attack to less than one day [104]. Although these examples provide powerful countermeasures and meaningful insights, they still do not fully leverage the interaction with an attacker. We propose to organize the deception technology into defensive and offensive deception technologies, where defensive are the technologies we described and offensive is leverages the attacker interaction to fight back. To the best of our knowledge, no existing work introduces this distinction yet and published surveys focus on defensive technologies only [105, 106, 107]. The benefit of offensive deception is that it allows us to monitor the attackers' activity beyond the target system. Conceptually, one could attach a tracker to the attacker and monitor where else it shows up. Monitored credentials have been used as such trackers and a manual deployment has shown that phishers commonly try to test their harvest after 1-3 days - giving investigators limited time to hunt them. Automated decoy credentials have been introduced by Humboldt 1.0 who proposed a distributed system to stuff phishing sites with fake credentials and monitor their usage. Humboldt 1.0 suffered from IP address blocklisting by the attacker [108], so its successor, Humboldt 2.0, leverages human participants to overcome this issue [109]. At a similar time, BogusBiter was published, a Browser Extension that dilutes any password submission with 4-12 additional credentials, such that a phisher cannot know which credential is the real one[110]. As a critical remark, none of these publications perform a real-world evaluation, which was only done recently in 2018 by Akiyama et al. [111]. We further notice the lack of technical sophistication, as in the assumption of non-encrypted communication and the use of plaintext passwords, as well as with regards to IP address rotation and handling cloaked sites. More importantly, these solutions focus on credential stuffing only and do not implement a solution to automatically submit any other information - but which would be crucial for stuffing PII Phishing sites. Lastly,

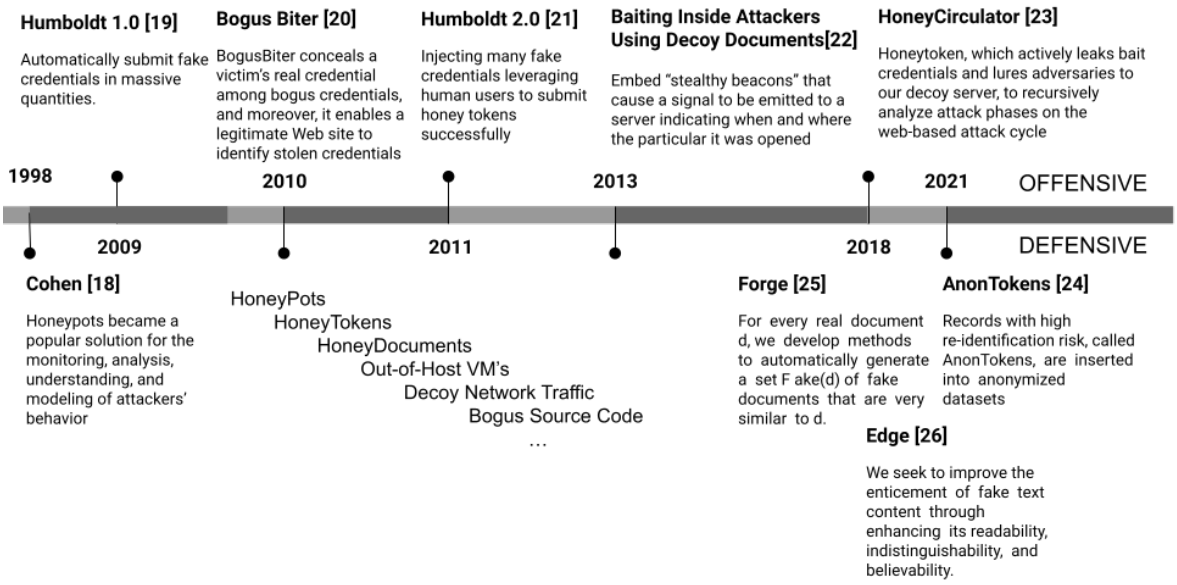


Figure 4.1: Timeline of related work in deception technologies.

phishers have matured and modern campaigns can be expected to verify their catch before retailing it. Therefore, any sort of infiltration has to happen in a believable manner.

4.2 On the Believability of Data

In everyday conversation, we refer to something as ”believable” when its essential features conform to our expectations, another word for it could be ”authentic”. For the sake of this work though, we use the term ”believable”, because the terminology around ”authenticity” and ”authentication” may create confusion for the reader as they refer to a distinct concept in cryptography. With this separation in mind, believability and authenticity conceptually describe the same idea: The information we generate should not tip off its reviewers as potentially fake or computer-

generated information. Believable information looks normal. The term ”normal” properly suggests the use of normal distributions of information, namely the frequency of an information instance occurring in the wild. Accordingly, we would gain maximum benefit by systematically enumerating a phisher's verification vectors and then deceiving each verifier with a set of information that follows the expected distribution. The following sections describe in detail how we generate believable information that comes from believable sources and can trick human reviewers (phishers) into false-positive verification. An attacker can easily evaluate the source of her catch based on meta-information that was generated as an artifact of the communication with the victim. Specifically, the Internet is implemented as a standardized network stack, described by the OSI model. Accordingly, a modern website is accessed by exchanging information on seven layers of communication: Application (HTTP), Presentation (TLS), Session (Port), Transport (TCP), Network (IP address), Data Link (MAC address), and Physical layer (fiber cable). While most of these layers are fixed and do not change for each communication partner, any communication partner can try to obtain and evaluate her peers' origin by looking at her IP address and application data. On the application layer, a web application can gather information about its visitor, specifically about the visitor's browser. Today, JavaScript implements the interface between a website and its environment and it may call a number of browser APIs, e.g., to access a list of cookies, the user's browsing history, open tabs, and the presently installed plugins. Snyder et al. enumerate modern JavaScript features and point at their surprisingly rare usage in the wild [112]. To the best of our knowledge, the two verification vectors - network and application layer - provide the phisher's only ways to assess the origin of an input. In the following subsections, we will systematically mitigate these verification vectors.

4.2.1 Network Layer

Phishing campaigns are deployed at different scales and different scopes, e.g. they can be configured to render differently in different regions of the world or not render in some areas at all. Phishers can filter IP address ranges to restrict the scope and to limit the access of (known)

web crawlers and threat analysts because they can expect their victims to access the phishing site from an ISP gateway. For example, it is highly unlikely that a victim is surfing the phishing site from cloud instances - unless the victim is routinely using a VPN. The phisher can easily recognize access from a cloud instance or commercial VPN since such IP address ranges are publicly known.

4.2.2 Network Layer Deception

To overcome IP address filtering, we rotate our source IP address, without using the cloud. Instead, we leverage collaboration with a medium-sized Internet Service Provider - Columbia University (AS14) - managing multiple Class C and Class B networks. Within these networks, we can switch our public IP addresses on demand, such that the phisher can hardly recognize us as the same visitor. We evaluate the access from three different networks: 1. Internet Service Provider AS14 2. Private Internet Gateway 3. Commercial VPN

4.2.3 Application Layer

Like any other website, phishing sites can see the victim's User-Agent as part of the HTTP request header. Modern browsers use this field to identify themselves, such that web applications can present the response according to the needs of the specific browser. However, it is only a convention to use the actual browsers-identifier as User-Agent and the text field is easy to spoof. In addition to the User-Agent, modern web applications evaluate and cross-correlate all sorts of information using JavaScript and browser APIs. While this level of user fingerprinting is a research field in and of itself, we identified that a handful of characteristics suffice to tell a real visitor from a crawler. Namely, a web application can evaluate the four browser components history, bookmarks, extensions, and cookies to evaluate if a visit is coming from an automated headless browser.

4.2.4 Application Layer Deception

To model a believable user environment we use Selenium WebDriver, a project originally meant to automate web application unit testing. Selenium WebDriver allows us to build a powerful bot

that dynamically interacts with any given website. We can indistinguishably model a browser environment by rotating the contents of four browser components (History, Bookmarks, Extensions, Cookies) and the User-Agent property, and then we can craft and submit the decoy information at a pace that is consistent with human behavior. From this fake browser, we then call the target site and provide it with the desired information. We use the selenium's sendKeys() API to enter the information, instead of directly pasting it. Each API keystroke's delay can be refined with a stronger model, like that of an empirical typing pace study [113, 114], but which we leave to future work.

4.2.5 Data Layer

A victim's personally identifiable information (PII) can be abused to emulate the victim and to commit a multitude of real-world crimes in his or her name. For instance, credit card information can be used in drug deals, Social Security Numbers and health ID's can be used in insurance fraud, and passport data can be used for human trafficking or terrorism. Even information that is thought of as "publicly available", such as a combination of name, birthday, and phone number, may be sufficient to register services and open new accounts under the victim's identity (unauthorized account creation). Accordingly, modern phishing campaigns have outlived the goal of account take-over and it is excelling towards more sophisticated, more dangerous targets. Our system handles all phishing; both credentials stealing and identity theft.

4.2.6 Data Layer Deception

We developed a fake information generator capable of producing sets of information that satisfy the majority (if not all) of information that modern phishers hunt for. We aim to draw all possible PII in a conclusive, believable manner. Namely, we approximate real-world probability distributions for that information to exist. For example, the prevalence of a specific ethnicity may vary based on a particular region of the world (see Network Layer Scope) and a name may be associated with ethnicity. Conversely, that name may indicate the birth year that it was most pop-

ular in and a person's birth year is ultimately reflected in his or her social security number. This example demonstrates how even seemingly independent information like ethnicity and SSN can be connected if additional information is used to link it. To avoid this, we create a stepwise process that derives one information from the other.

4.2.7 Testing / Verification Layer

In the common case of credential phishing, the acquired credentials may be tested. A phisher may perform a test login to a web account, or a network intruder may confirm that the credentials provide access to a machine. Either way, the manual verification process passes four checkpoints:

1. The account exists
2. The credentials grant access)
3. The look and feel of the account is legitimate
4. The account seems to have value

4.2.8 Verification Layer Deception

The first check will only be passed if the institution whose name was abused to run the campaign can host decoy accounts for this purpose. We declare arbitrary online accounts out of scope and focus on the stuffing of email addresses that point to a domain under our control - Columbia University's email server - and we host fake email accounts on it. As this mail server is a honeypot in and of itself, any external login attempt reveals the IP address of a phisher. Once login to our mail server succeeds we want to present a highly believable email inbox and outbox, with realistic conversations and meaningful email attachments. This will check the third box of information verifiability. This is necessary because empty or otherwise suspicious-looking mailboxes may alert the phisher as the sheer idea of honey accounts is not a novel one [115, 116, 117]. Additionally, the email account is seemingly functional, as it permits outgoing traffic, but the traffic is relayed and the relay is blocked by a firewall. We implement the indirection to avoid suspicion. As an

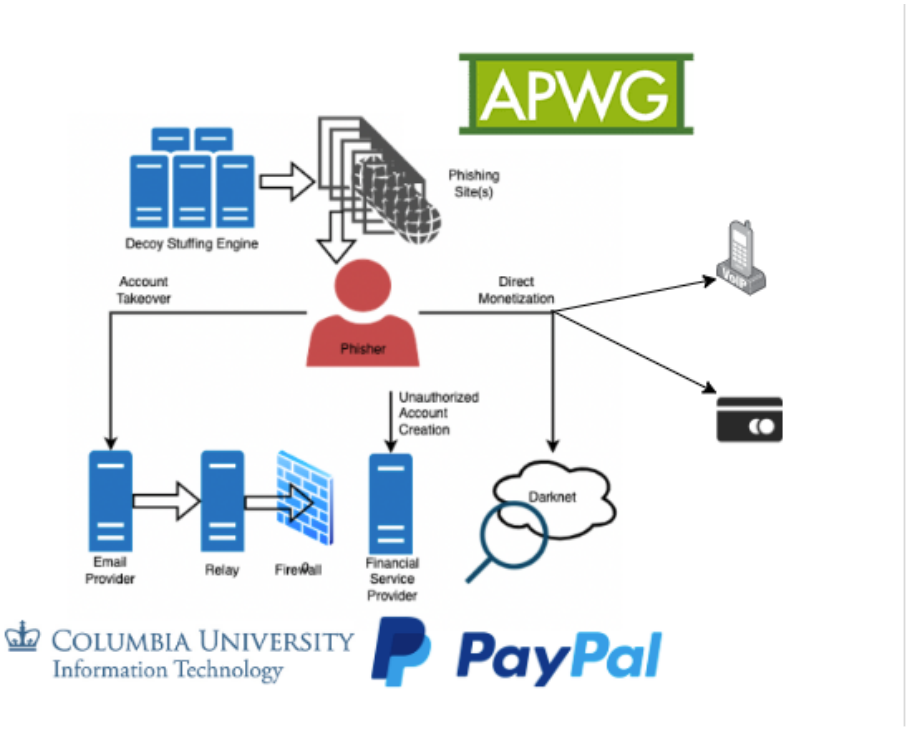


Figure 4.2: Installation of decoy sensors across the Internet.

alternative to this setup, which can be easily replicated with a self-hosted email server, we propose to use public email accounts and enable MFA to receive notifications for login attempts. Another alternative is the collaboration with a target brand, who may host fake accounts and monitor account activity. Such collaboration is particularly promising since it can now provide phishers with credentials to the exact service they are looking for. We explore this route in the next section.

4.3 Tracing Information Abuse

Interacting with cybercriminals is a promising undertaking, allowing us to improve our understanding of a threat and even attempt threat actor attribution. We propose to provide phishing sites with believable decoy information and to trick phishers - who were trying to trick us first - into using these decoys. To monitor the phisher's usage of our decoy on the Internet we set up a number of sensors that we can monitor to trace the phisher's activities.

We establish the following sensors for our experiments:

- Email Server Login
- (MitM Email Outbox)
- Target Brand Login
- Target Brand (unauthorized) Account Creation
- Third Party Black Market Monitoring
- Burner Phones (VOIP)
- Deactivated Credit Cards

This variety of sensors enables us to investigate if a phisher tests to login to the domain of an email credential, if she tests that the account exists on the victim brands site, if the phisher tries to create an account, if she directly monetizes the credit card, or if there's any other, potentially untested retail activity on the black market. Our collaborating third-party black market scanner is capable of scanning over one billion credentials a month, to the best of our knowledge one of the best of its kind. In addition to the black market scanner, we also establish phone numbers, and credit card numbers that the fraudster cannot use without us getting notified (more details in Chapter 8). Apart from the monitoring setup, we overcome the limitations of Humboldt 1.0 by deploying tracers from rotating IP addresses - and without relying on mechanical-turks. Further, we run dedicated counter-attacks instead of diluting an active password submission like BogusBiter did [110]. We are the first to submit PII, instead of only account credentials, which allows us to investigate the emerging threat of unauthorized account creation and to better understand why phishers increasingly target PII.

4.4 Denial-of-Phishing Engine

The ultimate goal is to implement a Denial-of-Phishing Engine (DOPE) that takes a phishing site as its input and automatically attacks it with automatically generated believable decoy PII that

it submits in a believable manner and from indistinguishable sources. We build such a system and keep it configurable to the insights of our comparative field study in Chapter 10. In addition, we develop a research frontend in the form of a browser extension, that can be used as an interface to the backend to easily fetch decoy PII and stuff a phishing site from a ISP gateway. The browser extension serves as a source IP diversifier and simultaneously allows DOPE to improve by collecting usage statistics. Leveraging the community we can ultimately collect more features and learn about what other information other phishing sites target. We discuss the details in the two upcoming sections.

4.4.1 Implementation

DOPE Engine The main backend component of the Denial-of-Phishing Engine (DOPE), the assembler❶, takes as an input a phishing URL ❷ from one of many possible sources, e.g. a proprietary scanner, public phishing feeds, or manually supplied through the frontend web application. The assembler then visits the target phishing site to learn its PII requests and it fetches the respective response from an extendable NoSQL database of pre-generated PII. In the backend, a generator ❸ ensures that the database is provisioned with sufficient PII. We do not generate the decoy ad hoc, because we need to ensure that our black market monitoring has sufficient time to learn which tracers to look out for. Once the decoy PII set is assembled, it gets forwarded to one of three attack relays ❹. We implement the attack relays as a docker container and deploy it across multiple systems in different networks, such that we can shoot tracers from our private internet gateway, from within the ASN we control, and from public cloud providers. To evaluate the case with no ASN access we also experiment with the use of foreign VPN services. **DOPE Browser Extension** Preliminary experiments suggest that spoofing believable IP addresses may become a bottleneck for this offensive deception approach, while the other information is straightforward to spoof, as we discussed. Subsequently, we learn from related work and propose to ”leverage the crowds” by deploying a browser extension that works on top of DOPE, but enables contributors to participate from their local internet gateway. The DOPE Browser Extension also provides feed-

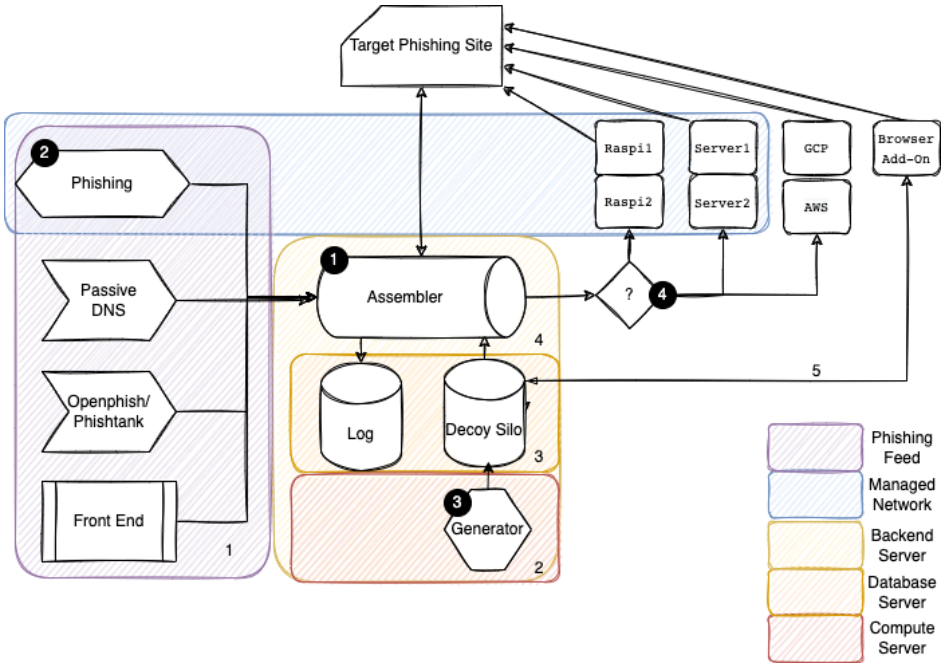


Figure 4.3: Overview of the Denial-of-Phishing Engine (DOPE).

back about sites that were attempted to attack and what PII these sites requested. It's written in JavaScript REACT and available open-source via GitHub.

4.5 Decoy Injection Parameter Comparison

This section provides a comparative analysis of different decoy provisioning methods, which can help us understand if and how phishers verify the authenticity of stolen data. We hypothesize that phishers verify their catches, which is motivated by two simple observations: first, verification can avoid detection by threat analysts, and second, verification can avoid polluting the catch with invalid information. Since DOPE embodies a system to pollute a phisher's catch with invalid information, it is paramount to our design that we understand and overcome the phisher's verification vectors. We discussed verification vectors in Section 4.2 and we will now provide an overview of the experimental setup that we use to A/B Test different data injection methods.

	Provider	IP address	Environment Logged in	AutoFill	Website	LinkedIn	Facebook	Paypal
Network	UNI	local	TRUE	FALSE	TRUE	TRUE	TRUE	TRUE
	UNI	foreign	TRUE	FALSE	TRUE	TRUE	TRUE	TRUE
	UNI	campus	TRUE	FALSE	TRUE	TRUE	TRUE	TRUE
Verification	GMAIL	local	TRUE	FALSE	TRUE	TRUE	TRUE	TRUE
	GMAIL	local	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE
	GMAIL	local	TRUE	FALSE	TRUE	TRUE	TRUE	FALSE
	GMAIL	local	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE
Autofill	GMAIL	local	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
	GMAIL	local	TRUE	TRUE	FALSE	FALSE	FALSE	FALSE
Auto Engine	FALSE	local	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE
	FALSE	local	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE

Table 4.1: Eleven configurations to investigate different outcomes in network- , verification- , and application layer (autofill) parameterization. The Auto Engine stuffs random decoy as a baseline.

4.5.1 Experimental Setup

We use a fixed set of ten pre-generated decoys, stuffing each of them with different parameters, enabling us to accept or reject four hypotheses about data-authenticity verification methods a phisher may use. We pick up the four layers of believability introduced in Section 4.2 and we compare different outcomes for each of them. Namely, we use different IP addresses to learn if the phishers test on the network layer, we create accounts with and without social media presence, as well as with and without an account at the targeted brand, to learn if the phisher tests on the data and verification layer. When filling in the information, we do and do not log in to the browser environment as well as we do and do not enable auto-filling, which we believe the phishers use to distinguish a victim from a web crawler on the application layer. Table 4.5.1 illustrates these configurations as a truth table. Therefore, each row in Table 4.5.1 represents one pre-generated set of believable fake information listed in Appendix E. At the abstract level, each set of this fake information constitutes an identity - an instance of PII - we use these fake identities to investigate same-day clones of paypal.com, sourced from the Anti-Phishing Working Group's eCrime eXchange (APWG eCX). Appendix E shows a full list of the URLs we reviewed. While most of the URLs on the phishing feed were already taken down by the hosting service or DNS registrar, we found between one and three active phishing instances of paypal.com every day; also marked in Appendix E. This distinction between active and non-active sites was made separately, before

starting the controlled experiment and from a different machine, such that we do not accidentally enable the phisher to recognize or track us, and such that we do not distort results with our preliminary analysis. We surfed the URL from the browser we work with on a daily basis using our private internet gateway, which is an IP address located in the Verizon Network in New York City. If a site was not available, we double-checked it using a mobile User-Agent - our personal iPhone 11 using its Verizon data plan. In both cases the actual IP address changes throughout the experiments, but which should not make a difference for these singular test accesses. We then conduct the data injection as a controlled experiment rotating between Windescripe Premium VPNs and the Columbia Network to appear as independent visitors and recording HTTP Archives (HAR) of the network communication with the target phishing site to enable post-mortem analyses of the sessions. During decoy injection, we monitor credit card transactions, and phone notifications in real-time, to detect whether phishers test the injected data. Unfortunately, we had to learn that the delay of credit card decline notifications is subject to the card issuer's load and may not be reflected on the UI in real-time. However, we believe that the slow pace of our manual interaction with the site still allows for an accurate mapping between the data injection and the phisher's misuse. We discuss this in the evaluation.

Next, we describe how we generated believable decoy data in practice for the following data classes:

- Personally Identifiable Information (PII)
- Social Security Number (SSN)
- Government ID
- Phone Number
- Account Credentials
- Mailing Address
- Credit Card

- Bank Account
- 3-D Secure Account
- Browser Profile
- Autofill
- HTTP Request Headers
- IP Address

Personally Identifiable Information (PII) We freely invent names and birhtdays covering a variety of suggested nationalities ethnicities and ages.

Social Security Number American social security numbers are most conveniently applied for at birth, and prior to 1972 the first three digits comprised the Area Number which revealed the issuing state, which was likely to be the location of birth. Then, between 1972 and 2011, SSN's first three digits were associated with the ZIP code of the application. The middle two numbers comprise the Group Number following a sequential pattern, so could be validated against the birth date. Since 2011 SSNs have been randomized. For this experiment, we leverage an online SSN generator that provides a fake SSN based on a given birth date and state (ssn-verify.com). With the different SSN algorithms in mind, we will use birthdates before and after 1972 and 2011 for this analysis (note that anyone born in 2011 considered of legal age today).

Government ID Similar to the SSN we do not bother generating these numbers ourselves and employ free online services instead (see `https://www.ssn-verify.com/generate` and `https://www.elfqrin.com/usssndriverlicenseidgen.php?ckpassport=1`). If phishing sites ask for passport photos we upload two different photos of the outside of a passport, hoping to pass the file-upload test and not to tip off the phisher with identical uploads. An overview of how many sites asked for this is in Table X.

Phone Number We employ real phone numbers using free VOIP number providers `https://www.textnow.com/` and `https://abine.com/`. However, we notice that benign ser-

ID	First Name	Last Name	Email	Birthday	Billing Address	Phone No.	SSN	Mother's Maiden Name
1	Wei	Zhuang	wz2567 @columbia.edu	04/12/1988	552 Mudd 500 W 120th St, New York NY 10027	(857) 5763-578	115-92-8858	Zhuang
2	Mark	Ferrante	mf3412 @columbia.edu	07/20/1982	547 Mudd, 500 W 120th St, New York NY 10027	(631) 980-1913	075-35-3432	Ferrante
3	Zeynep	Aksoy	za2297 @columbia.edu	05/03/1977	518 Mudd 500 W 120th St New York NY 10027	(914) 296-0337	523-25-7855	Aksoy
4	Pete	Klevenstein	pete.klevenstein @gmail.com	05/03/1991	1029 Sip Ave Medford, 11763	n/a	1992-08-14	Kleven-stein
5	Clara	Rubens	clararubens98 @gmail.com	11/28/1998	120 West 50th Street Apartment 7H	(312) 300-9984	090-62-5568	Rubens
6	Maria	Anna	mail.maria.anna @gmail.com	07/21/1981	342 Broadway Apartment 423 10011, NY	(631) 318-6054	130-76-9215	Anna
7	Joseph	Dilama	dilamajoseph60 @gmail.com	09/23/1960	123 Greene St New York NY 10245	n/a	054-68-5640	Dilama
8	Juri	Wizlaw	juri.wizlaw @gmail.com	04/02/1965	333 East 65th Street Apt 10 New York NY 10065	(631) 1315-2430	065-56-1245	WIzlaw
9	Destiny	Campos	hardcoredestiny69 @gmail.com	05/18/1992	88 W 14th St, New York, NY 10011	(914) 486-5440	076-72-2137	Campos

Table 4.2: Overview of the PII we use to organize the decoys as "identities".

vices like email providers or PayPal itself recognize such numbers as VOIP numbers and reject them. We suspect that phishers have the same capabilities since we didn't receive any calls or text messages on our 18 VOIP phone numbers. We focus on this disappointment and register three real mobile phone numbers in the T-Mobile network. The three numbers should be indistinguishable from real phone numbers with the only limitations that all three use the area code *+1 (646)* for Manhattan.

Account Credentials We generate real email accounts at popular providers such as Microsoft Outlook, Yahoo! and Gmail. The password to these email accounts is secret and is not shared during any of our experiments, but we enable two-factor authentication so that login attempts send a notification to our smartphone. We then use these email accounts to create real PayPal accounts with different passwords. Those are the passwords we will ultimately share with the phishers. Before we share the credentials, PayPal flags the account as compromised and enables internal monitoring for us. We do not link a credit card, bank account SSN, passport number or any other additional information to the accounts and we do not enroll in any of PayPal's additional services such as "Pay in 4" and alike. A phone number is registered for account recovery, but two-factor

authentication is disabled (as it is by default).

Mailing Address For each identity, we choose random existing addresses in geographical proximity to one of our VPN provider's servers so the shared ZIP code and IP location are consistent. Mismatched geolocation is an easy test used by some phishers to validate a victim's identity is suspicious or not authentic.

Credit Cards Debit or credit cards embed a physical chip and/or magnet stripe to pay in so-called Card Present transactions (CP). However, the 16-digit number, expiration date, and CVV2 code suffice to place an order remotely, called a Card Not Present transaction (CNP). Subsequently, the triplets have the value of the card's available credit limit and thus they make an obvious target for criminals. To generate fake instances, we can use a random CVV2 number, an arbitrary expiration date less than four years into the future, and a card number that passes the standard credit card verification algorithm, the Luhn checksum. However, real credit cards also incorporate the issuing bank's Bank Identification Number (BIN) within the first 4–6 digits of the card number and so the Luhn conform number may not pass payment authorization in practice and it may end up being an early show stopper for us. Therefore, we leverage the virtual credit card services provided by Privacy.com where we create a virtual credit card for each fake identity in our experiment. We load the credit card from a bank account with the minimum required amount of 1 USD and we then pause the card, such that any transaction attempt will be declined. This further enables the detection of transaction attempts in our virtual credit cards' payment log. An additional benefit of these cards is that they allow the user to arbitrariliy choose a different name and billing address for each transaction, which gives us the flexibility to choose mailing addresses based on VPN locations as described earlier. As a side effect of using Privacy.com, every card is has the same identical BIN *453641* and expiration date *04/2028*, which enables a quick search for black market leaks of the stolen credit card.

Bank Accounts To pass any correlation with the credit card number's BIN, we enter all bank information as if it would belong to Card Services for Credit Unions, using their actual routing number *321177735*. We then add an account number using Python package Faker and add a

ID	First Name	Last Name	Email	Credit Card No.	Exp. Date	CVV
1	Wei	Zhuang	wz2567@columbia.edu	4536410089866111	04/28	236
2	Mark	Ferrante	mf3412@columbia.edu	4536410131666188	04/28	534
3	Zeynep	Aksoy	za2297@columbia.edu	4536410037509631	04/28	931
4	Pete	Klevenstein	pete.klevenstein@gmail.com	4536410043508023	04/28	343
5	Clara	Rubens	clararubens98@gmail.com	4536410185578214	04/28	857
6	Maria	Anna	mail.maria.anna@gmail.com	4536410066473998	04/28	285
7	Joseph	Dilama	dilamajoseph60@gmail.com	4536410073833887	04/28	920
8	Juri	Wizlaw	juri.wizlaw@gmail.com	4536410121472043	04/28	176
9	Destiny	Campos	hardcoredestiny69@gmail.com	4536410024932077	04/28	908

Table 4.3: Overview of the individual decoy identity's credit card information.

random four-digit number as ATM PINs. We note that correlations with real bank accounts are possible but unlikely given the length of the account number plus its use in combination with a customer name.

3-D Secure Strong customer authentication methods like Verified by Visa, MasterCard Secure Code, and American Express SafeKey are commonly implemented by retailers to comply with regulations defined in the EU's Revised Directive on Payment Services (PSD2), and to reduce credit card fraud. We note that phishers have picked up on the trend and in many cases ask for 3-D Secure passwords in combination with the credit card triplet. In this case, we supply the same passwords we used for the respective PayPal accounts, which is believable because real people tend to reuse passwords.

Browser Profile In addition to the rather obvious mapping of a visitor to an IP address, the visitor of a website can be fingerprinted by browser characteristics. Browser fingerprinting goes beyond the User-Agent used in the HTTP request header that we discuss later. Instead, it also commonly leverages the site's access to open tabs, installed extensions, bookmarks, browsing

ID	First Name	Last Name	Email	Bank Name	Account Number	Routing Number	ATM PIN
1	Wei	Zhuang	wz2567@columbia.edu	Chase	1234567890	987654321	8765
2	Mark	Ferrante	mf3412@columbia.edu	Wells Fargo	70744737	165653787	6578
3	Zeynep	Aksoy	za2297@columbia.edu	Credit Union	85190836	321177735	9898123
4	Pete	Kleven-stein	pete.klevenstein@gmail.com	Wells Fargo	76848309	179377635	0111
5	Clara	Rubens	clararubens98@gmail.com	Credit Union	26088541	321177735	3257
6	Maria	Anna	mail.maria.anna@gmail.com	Credit Union	799719505	321177735	4654
7	Joseph	Dilama	dilamajoseph60@gmail.com	Credit Union	87299269	321177735	2901
8	Juri	Wizlaw	juri.wizlaw@gmail.com	Credit Union	31127896	321177735	5656
9	Destiny	Campos	hardcoredestiny69@gmail.com	Credit Union	23292509	321177735	6162

Table 4.4: Overview of the individual decoy identity's banking information.

history, cookies, and more. A phishing site may use browser fingerprinting to test the authenticity of a victim, so we try to provide a believable browser profile that the phisher may accept to be a real victim. We use a default Google Chrome, the Internet browser with the highest market share today and we do not add any bookmarks or extensions to it, which we reason to be realistic enough. We create a short browser history of social media sites and search engine queries and persist the cookies they set, but we remove any additional cookies after each experiment, to ensure the phishing site does not persist a session cookie, meaning it could readout that session cookie and recognize us across experiments. In addition, we open four to six tabs during each experiment. These are the email provider, paypal.com, textnow.com, google.com, and if the decoy set is configured to "have social media accounts" we also open twitter.com, facebook.com, and linkedin.com where the browser is logged in into all of them. In case the phisher tests for active logins, we pass this test.

Autofill There are several ways to provide input to a website. Users may type inputs using the keyboard, use the browser's auto-fill function, use a dedicated password manager, or copy paste the information from elsewhere. A careful phisher may try to evade web crawlers or threat analysts by verifying input methods are from human users rather than programs. We implement a simple test to learn if phishers in the real world test how data is provided: For some accounts, we enable the browser's auto-fill feature and use it to submit name, email, password, address, and payment information, because to the website it looks the same as pasting it from somewhere. Although this is an incomplete test, it covers the main case we are interested in with respect to the automation of DOPE.

HTTP Request Headers Per the default of Chrome Version 101.0.4951.54 (Official Build) (x86_64) on MacOS BigSur Version 11.6.1 the User-Agent is "Mozilla/5.0 (Linux; Android 6.0; Nexus 5 Build/MRA58N) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/101.0.4951.54 Mobile Safari/537.36". We decide to leave it untouched to appear as an authentic MacOS user. However, we note that some phishing sites immediately redirect our requests to benign sites, so we go over them a second time switching our User-Agent to the chrome browser's default User-Agent

ID	Name	Description	IP Range (example)
1	**Wei**	Verizon Internet Gateway	24.90.102.. . .
2	**Marc**	Frankfurt, Germany Windscribe VPN	87.249.132.. . .
3	**Zeynep**	Columbia University CRF VPN (ASN14)	128.59.13.. . .
4	**Pete**	New York City, NY Windscribe VPN	217.138.255.. . .
5	**Clara**	New York City, NY Windscribe VPN	217.138.255.. . .
6	**Maria**	New York City, NY Windscribe VPN	217.138.255.. . .
7	**Joseph**	New York City, NY Windscribe VPN	217.138.255.. . .
8	**Juri**	New York City, NY Windscribe VPN	217.138.255.. . .
9	**Destiny**	New York City, NY Windscribe VPN	217.138.255.. . .

Table 4.5: Overview of the individual decoy identity's IP addresses.

for mobile debugging: "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/101.0.4951.54 Safari/537.36". In both cases the accepted-language header is "en-US,en;q=0.9,de;q=0.8", meaning that US English and German are accepted languages and preferred with a respective weight of 0.9 and 0.8.[1]

IP Address We use three distinct IP addresses to stuff the PII from. First, we use a Verizon gateway with the IP address 24.90.102.. . . , second, Windscribe Premium VPN's "Empire State" location in downtown Manhattan, assigns us with rotating IP addresses in the 217.138.255.. . . range, and we also use a Germany based address (87.249.132.. . .) to test for geolocation filters. Lastly Columbia University's network allows us to rotate IP addresses without relying on a well-known commercial solution, and we hope that with 128.59.13.. . . , we fly under the radar for longer.

4.5.2 Real World Findings

Now that we have described the decoy data generation and choices made to produce believable and authentic appearing data, we next describe the formal evaluation of the efficacy of this decoy data in passing phishing website filters.

We manually reviewed a total of 28 phishing sites impersonating paypal.com and infiltrated them with decoy PII crafted according to the above explanations. Although almost none of the

[1] German as a second language was not a conscious choice and may be used against us by the phisher

clones looked identical we note similarities between the landing pages and user journeys they implement. The landing pages asked for either email and password simultaneously or led from email to password using a JavaScript effect. We consider this difference to be only visual since the use of JavaScript and our network trace review suggest no data submission to the server. Only one site mimicked the real PayPal landing page and expected the user to click the login button to get to the login form. For the rest of the victim's journey through the phishing application, we observed a pattern of information requests in a specific order. Most phishing sites start by lulling the victim with a page that promotes the detection of "suspicious behavior" and the necessity to confirm one's identity. The malicious site then proposes to do so by requesting credit card information, a billing address, 3-D Secure credentials, bank account information, and finally uploading a photo of the victim's government ID or passport, in this particular order. Respectively, our findings result from the victim's separate steps through the phishing application, and from our various sensors described earlier in Section 4.3.

Findings from the Phishing Landing Page

At this initial step of the user journey, the login page, we observe that almost all phishing sites yield an error message if we use a username that is not registered on paypal.com, or if we use a wrong password. Only credentials that belong to actually registered PayPal users get past the login form, where the phishers then ask for additional information on top of username and password. We conjecture that phishing sites test credentials in real-time, which is confirmed by the text message notifications sent out to us by the victim brand, PayPal, who instantaneously detects suspicious behavior and alerts the victim user. PayPal's log files confirm these accesses and indicate that those accesses are simply login attempts to test the credentials. Only in one instance did the phishers try to enroll the stolen account to PayPal's loan-based "Buy Now Pay Later" program *Pay In 4*, but which was correctly rejected by the platform. Generally, though, the fraudsters do not inspect the account any further or make changes to it. We found this particularly interesting because many phishers ask for the mailing address, phone number, SSN, and passport number, redacted versions

of some of which are visible under PayPal's account settings tab, but which the phishers do not seem to confirm this way. Since we cannot know the true reason why a phisher would miss this opportunity we simply reason that either credential verification is sufficient, or the possibility had not occurred to the phisher, or that this way of verifying data is not valued in the underground.

Findings from the Phishing Application

The last step of the user journey, the file upload for a government ID and passport deserves special attention. It is an obvious way for the phisher to catch an additional piece of information, that is valuable in and of itself, and can also be used to verify the rest of the information. Although file upload can be implemented easily, it is an additional obstacle for web crawlers, like DOPE and adding this capability will be a crucial design consideration. We respect the importance of this feature and study the logic of phishing sites under four cases of human user behavior:

Does the phisher process the stolen information differently if the victim opts out after the initial login, at file upload,or if the victim uploads two arbitrary, non-government ID files, like nature photos, or if the victim uploads two identical files? Unfortunately, all every case we tried was handled identical and we could not find any differences, yet.

Findings from Credit Card Sensors

On top of the victim brand's access logs and text message notifications, we also review the billing statements of the deactivated credit cards that we stuffed into the phishing sites. While all transactions were declined, the billing statements still show the timestamp, dollar amount, and merchant. The next section describes two representative sets of credit card transactions that the phishers initiated on our cards and discusses the respective insights.

One phishing site tried using credit card numbers within minutes after the submission to the site, making the six payments at three merchants listed in Table X. Every account that we stuffed and that actually exists on paypal.com was hit by this attack, except for one. We later find in the logs that we had a typo in the missing decoy victim's email address, which explains the discrep-

ancy.

We research the merchant names and we learn that two of the three merchants are popular and well-established computer hardware stores in Germany which matches the language used by the phishing site - German. The third merchant "Otpmobl*sziget.hu" is associated with a Hungarian music festival. We contacted the three merchants on the same day of the transactions, asking them for any insight they can share about the purchase. The sites do not share a contact for fraud reporting, so we sent emails to the general customer service address asking about the name and address used for the purchase. The two German contacts responded immediately, confirming that the charges were made on their sites, but unable to provide further insight due to data protection laws. The Hungarian music festival however does not respond to our email or follow-up email. We note that the dollar amount of the transactions was identical for each payment at a given merchant and it is a considerably high value that would likely catch the card owner's attention, but still low enough not to be considered a high-risk transaction by PSD2, which sets a threshold of $1,200 to enforce strong customer authentication (3-D Secure). When looking for a pattern in the phishers' use of merchants, the timestamps almost suggest that the criminals move from one merchant to the other over time, but credit card A (see Table 4.6) is tested at two merchants and in an unexpected order that violates this pattern. Instead, it seems more likely that the phishers try purchasing at the merchants in round-robin order, or even at random.

A second phishing site also tried using credit card numbers within minutes after we shared them, but this time using a different approach, illustrated in Table 4.7. This time, only two accounts were hit, not all. The commonality between the decoy identities whose credit cards were picked

ID	Merchant on credit card statement	Value	Timestamp
1	**COMPUTERUNIVERSE.NET**	$391.46	10:35 am Card A (Wei Zhuang)
			10:33 am Card A (Wei Zhuang)
2	**Otpmobl*sziget.hu**	$586.13	2:18 pm Card B (Clara Rubens)
			2:37 pm Card C (Juri Wizlaw)
3	**Media Markt**	$623.16	3:56 pm Card D (Zeynep Aksoy)
			3:58 pm Card E (Maria Anna)

Table 4.6: List of transactions made by a real phisher using real credit cards a real merchants. The phisher attempted multiple transactions of over $500 each

up by the phisher was their IP address location. Both cards were stuffed from a local New York City-based IP address, one provided by Verizon, and one from the Columbia University campus network. The other decoy's identities, that the phishers did not use, were partially stuffed from IP addresses across the United States, and also New York City, but using a VPN provider, which we purposefully did to see if the PII gets treated differently by the phisher depending on its source IP address. In particular, we used the Windescribe VPN service, which is a known service and may have been recognized by the phisher. In addition, this phisher is more careful when testing the cards. The phisher leverages New York City-based merchants that roughly correspond with the source IP we stuffed from or the ZIP code we provided with the cards. In our personal opinion these three merchants are quite unusual. One is a food market on John Street, which uses mercato for delivery but does not seem to have a way of setting up accounts. And next level martial arts does not have any apparent web commerce. So how were these charges effected? Although neither the card issuer nor the merchant can give us the answer, we know that the payments happened fast, so we can outrule that someone printed physical cards from the data. An more realistic scenario may be that the phishers use malware to remote control compromised payment terminals. We have contacted the merchants via email.

On top of that, the zero-dollar charges we see on these statements indicate that the cards were added to the respective merchant website as new payment methods. When adding a card for later use, merchants commonly authorize the cards via a zero-dollar charge. These charges are commonly not listed on a credit card statement, and this way of testing may go undetected by the card owner - if the card is active. Note that this is different from the benign method that some platforms use to authenticate the customer when they send a small amount of money and ask to confirm the amount before reversing the transaction.

Linking phishing sites to attempted credit card transactions revealed two credit card verification and direct monetization patterns - valuable insight in the fight against fraud. We also draw conclusions about the phisher's modus operandi. The A/B Testing of different source IP addresses shows that phishers block VPNs and do filter the geographic location of an IP address.

ID (cont'd)	Merchant on credit card statement	Value	Timestamp
4	**Jubilee Market on J**	$0.00	9:53am Card E (Mark Ferrante) 9:53am Card D (Zeynep Aksoy)
5	**APPLE SLICES**	$0.00	9:53am Card E (Mark Ferrante)
6	**NEXT LEVEL MARTIAL ARTS**	$0.00	9:53am Card D (Zeynep Aksoy)

Table 4.7: List of transactions made by a real phisher using real credit cards a real merchants. The phisher attempted multiple transactions of over $0 each

Toward of Social Media Presence

In the last section, we discussed that randomly generated data will not pass most phishing sites' login form, because phishers instantaneously verify the login credentials at the target site, here PayPal. We conclude that decoy instances must have legitimate credentials for the victim platform. However, we also tested for differences caused by the victim user's presence on social media and we could not measure any indication that these phishers authenticate a real human user by testing their profile at other sites. We conclude that these phishers do not manually review the decoy PII's authenticity, because the phisher's login attempts at the victim brand are instantaneous, and the credit card transactions we monitored were timely, too. We did not observe a measurable delay that would have allowed them to e.g. search the victim's name on Google Search. After three weeks, we still did not register any action. Apart from manual review, there are ways to test whether or not a victim is currently logged in to social media sites when visiting the phishing site. Although this approach is hacky, it is broadly used for online marketing: A site can query another site's favicon without violating the same-origin policy. In the case of social media sites the availability of the favicon reveals the user's login status. To the best of our knowledge, this is the only way to operate such a lookup and we did not find any evidence for this technique in any of our experiments.

Autofill

The manual review of phishing sites taught us that some fraudsters disable the auto-fill feature, and pasting to the input fields. This seems particularly common to prohibit the user from pasting into the credit card number field. Henceforth, DOPE must implement data deployment via

JavaScript sendKey() API to handle these sites, which could handle all sites. On sites without this restriction, we did not measure a different outcome for decoy info provided via auto-fill versus such decoy info manually typed into the keyboard - they were processed equally by the phishers.

Through the controlled use of differnet parameters it was possible for us to better understand technical challenges for the automation of DOPE and we learned how some phishers orchestrate their credit card theft operations. This field study confirms that PII based sting operations are possible, and will provide valuable insights, up to an evidence of a crime, if used correctly.

4.5.3 Conclusion

In the previous section, we presented the hypothesis that phishers verify the authenticity of a victim and we described the design of comparative experiments to empirically test instances of the hypothesis. The experiment's evaluation shows several findings that this section will summarize and use to deduct guidance for the design of our Denial-of-Phishing Engine. We then conclude with additional recommendations for future fraud prevention.

It is important to understand that we do not hope to mitigate the entirety of a phisher's possibilities, instead, we use this study to learn which technologies are evident and we propose to scale DOPE to those. Sophisticated phishers may use additional verification vectors that we did not discuss or did not investigate as part of our experiments, but handling corner cases is not the ambition of DOPE, similar to how we do not consider spear and whale phishing but focus on spray'n'pray strategies only. It is said that the true threat of phishing is its low entrance barrier and this is where we aim to tackle, we lift the bar for script-kiddies, lower the financial incentive and increase the risk of legal persecution. Although the sample size of our field study is rather small, we emphasize that Chapter 4 showed how many phishing instances can be mapped to the same few phishing kits. Hence, we expect that few observations scale to many instances of phishing that use the same technology stack. Given the framework that this work provides, one can always add to it by investigating additional sites from additional phishing feeds. Similarly, we did not find any of our PII retailed on the black market, which may be due to the quality of our scanner, or because our PII

was dropped at a later stage, e.g. after the credit cards were declined. Regardless, our insights can provide valuable insights to threat hunters and automation of stuffing and credit card monitoring will be even more valuable. An instance of DOPE that is run by a payment network provider could deploy decoy PII and monitor usage ad hoc, then track and prosecute the threat actors. Similarly, the correlation of fraudulent payments can be used to inform merchants about their role in the system. Merchants lose transaction fees, time, and reputation when cybercriminals abuse them for credit card verification. Certainly knowing that a purchase was initiated with a stolen credit card provides legal evidence against the fraudulent customer, and the respective prosecution campaign may reveal additional intelligence about the fraud. For example, only the merchant can trace if the purchase stemmed from a registered account, if that account may have a shipping address associated with it, and if that address is associated with other accounts, too, or reoccurs in other fraudulent transactions. It may as well be that hackers deployed an Advanced Persistent Threat (APT) in the merchants' IT infrastructure and a digital forensics campaign may reveal such threat and prevent further harm. The possibilities are endless, the merchant may be in on it, and the phisher may use trojans to remote control end-users' computers to make the purchases.

The most critical design consideration of DOPE is not yet related to its data provisioning model. Most importantly we need the stuffed user credentials for any site to be valid for that site. This has been common among all phishing sites we got feedback from and at the same time, we consider this the most challenging and limiting aspect to overcome in practice because it inherently requires the a priori knowledge of the victim brand and a collaboration with that brand. Another consideration for the generated PII is for the ZIP code used for the decoy's address to match the source IP's geolocation.

To handle most phishing sites it is helpful to support a mobile User-Agent. In most cases, it is sufficient to spoof the User-Agent, but we suggest also to resize the window to a contemporary's smartphone screen's resolution. Similarly, DOPE shall spoof the HTTP header field for the referrer to it an email provider, such that the phishing site can see that "we clicked on a link". When navigating through the phishing app it is paramount to support cookies. In addition, it is desirable

to enter the decoy information via sendKey() API and to dacilitate file uploads. The credentials presence on social media platforms is secondary.

Conclusion

In this dissertation, we investigate the rising threat of PII Phishing. First, we consider the Detection, Triage, and Attribution of the threat. Then, we go one step further and we implement a system to automatically flood PII Phishing sites with fake information. We show that the fake data gets picked up by the phisher and it can be used to facilitate sting operations against the criminals behind the threat. At the baseline of this work, we observe that recent work on phishing detection is skewed towards developing new machine learning models. We also consider training a machine learning model to detect the threat, but in contrast to most publications, we do not focus on the architecture of the machine learning model. Instead, we explore the use of new features. New features provide a great value to the community since any future model can easily adopt them as part of their own feature set. We note that the community shares several numeric datasets, but feature engineering would not have been possible without a dataset of phishing page source, so we invite the community to follow in our footsteps by using our publicly available dataset of raw phishing page sources. In Chapter 2.4.5 we show the two new features yield promising insights. First, we consider training a machine learning model to detect phishing. Comparing the feature importances shows that our two new features can help to push the accuracy from 68% to 92%. Second, the features can be used to map a phishing site to a threat class, using a list of 356 PII indicators that we associated with one of 36 information categories. These information categories can further be used as a PII reference list to meet government regulations for data protection. Third, we propose to interpret the new features as phishing site fingerprints that leak the relationship between multiple phishing sites. A sample use case is the organization

of threat intelligence into a social network of phishers.

In Chapter 2.4.5 we present a framework for the automated interaction with PII Phishing sites. Our system is the first to automatically inject automatically generated PII into a phishing website and it is the first to consider the provisioning from believable sources such that an informed phisher will have a hard time distinguishing the decoy from a real victim. We show that the system can work in practice and we report the results from our fourteen-day interaction with twenty-eight phishing sites. The results show that some of our believability concerns are more justified than others and as a consequence , the system has great potential to be used at scale.

References

[1] *"The stockings were hung by the chimney with care"*, RFC 602, Dec. 1973.

[2] L. M. T. Berners-Lee and M. Mccahill, "Uniform resource locators (url," in *University of Minnesota*, 1994.

[3] *404 - CIA*, [Online; accessed 18. May 2022], May 2022.

[4] M. Hilbert and P. López, "The world’s technological capacity to store, communicate, and compute information," *Science*, vol. 332, no. 6025, pp. 60–65, 2011. eprint: https://www.science.org/doi/pdf/10.1126/science.1200970.

[5] S. Sjouwerman, *Phishing Remains the Most Frequent Attack Vector Used for Initial Access*, [Online; accessed 20. May 2022], May 2022.

[6] *Chinese Hackers Charged in Equifax Breach*, [Online; accessed 20. May 2022], Sep. 2020.

[7] A. Oest, Y. Safei, A. Doupé, G.-J. Ahn, B. Wardman, and G. Warner, "Inside a phisher's mind: Understanding the anti-phishing ecosystem through phishing kit analysis," *IEEE eCrime*, 2018.

[8] APWG, "Apwg phishing activity trends report," 2020.

[9] H. Tu, A. Doupé, Z. Zhao, and G.-J. Ahn, "Users really do answer telephone scams," in *28th USENIX Security Symposium (USENIX Security 19)*, Santa Clara, CA: USENIX Association, Aug. 2019, pp. 1327–1340, ISBN: 978-1-939133-06-9.

[10] A. Oest, *Leveraging Scalable Data Analysis to Proactively Bolster the Anti-Phishing Ecosystem*, [Online; accessed 20. May 2022], 2020.

[11] *Chinese Hackers Charged in Equifax Breach*, [Online; accessed 18. May 2022], Sep. 2020.

[12] S. Mohurle and M. M. Patil, "A brief study of wannacry threat: Ransomware attack 2017," *International Journal of Advanced Research in Computer Science*, vol. 8, pp. 1938–1940, 2017.

[13] M. Wu, 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*, ser. CHI '06, Montréal, Québec, Canada: Association for Computing Machinery, 2006, 601–610, ISBN: 1595933727.

[14] C. Whittaker, B. Ryner, and M. Nazif, "Large-scale automatic classification of phishing pages," in *NDSS '10*, 2010.

[15] S. Mohurle and M. M. Patil, "Phishing attacks and countermeasures," *Handbook of Information and Communication Security. Springer*, 2010.

[16] A. van der Heijden and L. Allodi, "Cognitive triaging of phishing attacks," in *28th USENIX Security Symposium (USENIX Security 19)*, Santa Clara, CA: USENIX Association, Aug. 2019, pp. 1309–1326, ISBN: 978-1-939133-06-9.

[17] P. Kumaraguru *et al.*, "School of phish: A real-world evaluation of anti-phishing training," Jan. 2009.

[18] E. Ulqinaku, D. Lain, and S. Capkun, "2fa-pp: 2nd factor phishing prevention," *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks*, 2019.

[19] [Online; accessed 18. May 2022], Nov. 2012.

[20] M. Matsuoka, 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 *2013 IEEE 37th Annual Computer Software and Applications Conference Workshops*, 2013, pp. 587–592.

[21] *Google Safe Browsing – Google Transparency Report*, [Online; accessed 18. May 2022], May 2022.

[22] [Online; accessed 20. May 2022], Apr. 2022.

[23] Internet Systems Consortium, *Internet Domain Survey*, [Online; accessed 18. May 2022], Aug. 2019.

[24] *Shodan*, [Online; accessed 20. May 2022], May 2022.

[25] *Historical list of new gTLD domain name registrations and zone deletions*, [Online; accessed 20. May 2022], May 2022.

[26] B. Wardman, G. Warner, H. McCalley, S. Turner, and A. Skjellum, "Reeling in big phish with a deep md5 net," *Journal of Digital Forensics, Security and Law*, 2010.

[27] J. Vargas, A. C. Bahnsen, S. Villegas, and D. Ingevaldson, "Knowing your enemies: Leveraging data analysis to expose phishing patterns against a major US financial institution," in *eCrime Researchers Summit, eCrime*, vol. 2016-June, IEEE Computer Society, 2016, pp. 52–61, ISBN: 9781509029228.

[28] S. N. Bannur, L. K. Saul, and S. Savage, "Judging a site by its content: Learning the textual, structural, and visual features of malicious web pages," *AISec'11 : proceedings of the 4th ACM Workshop Security and Artificial Intelligence*, p. 116, 2011.

[29] P. Dewan, A. Kashyap, and P. Kumaraguru, "Analyzing social and stylometric features to identify spear phishing emails," vol. 2014-January, IEEE Computer Society, 2014, pp. 1–13.

[30] S. Abdelnabi, K. Krombholz, and M. Fritz, "Visualphishnet: Zero-day phishing website detection by visual similarity," Aug. 2019.

[31] A. Caliskan-Islam *et al.*, "De-anonymizing programmers via code stylometry," *USENIX sec*, pp. 255–270, 2015.

[32] A. Caliskan-Islam *et al.*, "When coding style survives compilation: De-anonymizing programmers from executable binaries," p. 16, 2015.

[33] *: The Input Label element - HTML: HyperText Markup Language | MDN*, [Online; accessed 20. May 2022], May 2022.

[34] E. Lastdrager, I. C. Gallardo, P. Hartel, and M. Junger, "How effective is Anti-Phishing training for children?" In *Thirteenth Symposium on Usable Privacy and Security (SOUPS 2017)*, Santa Clara, CA: USENIX Association, Jul. 2017, pp. 229–239, ISBN: 978-1-931971-39-3.

[35] B. Reinheimer *et al.*, "An investigation of phishing awareness and education over time: When and how to best remind users," in *Sixteenth Symposium on Usable Privacy and Security (SOUPS 2020)*, USENIX Association, Aug. 2020, pp. 259–284, ISBN: 978-1-939133-16-8.

[36] E. J. Williams, J. Hinds, and A. N. Joinson, "Exploring susceptibility to phishing in the workplace," *International Journal of Human-Computer Studies*, vol. 120, pp. 1–13, 2018.

[37] W. J. Gordon *et al.*, "Evaluation of a mandatory phishing training program for high-risk employees at a US healthcare system," *Journal of the American Medical Informatics Association*, vol. 26, no. 6, pp. 547–552, Mar. 2019. eprint: https://academic.oup.com/jamia/article-pdf/26/6/547/34153669/ocz005.pdf.

[38] B. Kim, D.-Y. Lee, and B. Kim, "Deterrent effects of punishment and training on insider security threats: A field experiment on phishing attacks," *Behaviour & Information Technology*, vol. 39, no. 11, pp. 1156–1175, 2020. eprint: https://doi.org/10.1080/0144929X.2019.1653992.

[39] A. Oest *et al.*, "Sunrise to sunset: Analyzing the end-to-end life cycle and effectiveness of phishing attacks at scale," in *29th USENIX Security Symposium (USENIX Security 20)*, 2020.

[40] M. Campobasso and L. Allodi, "Impersonation-as-a-service: Characterizing the emerging criminal infrastructure for user impersonation at scale," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '20, Virtual Event, USA: Association for Computing Machinery, 2020, 1665–1680, ISBN: 9781450370899.

[41] B. M. Bowen, P. Prabhu, V. P. Kemerlis, S. Sidiroglou, A. D. Keromytis, and S. J. Stolfo, "Botswindler: Tamper resistant injection of believable decoys in vm-based hosts for crimeware detection," in *International Workshop on Recent Advances in Intrusion Detection*, Springer, 2010, pp. 118–137.

[42] C. Yue and H. Wang, "Bogusbiter: A transparent protection against phishing attacks," *ACM Transactions on Internet Technology (TOIT)*, vol. 10, no. 2, pp. 1–31, 2010.

[43] M. Husák and J. Cegan, "Phigaro: Automatic phishing detection and incident response framework," in *2014 ninth international conference on availability, reliability and security*, IEEE, 2014, pp. 295–302.

[44] E. Alowaisheq, "Cracking wall of confinement: Understanding and analyzing malicious domain takedowns," in *The Network and Distributed System Security Symposium (NDSS)*, 2019.

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

[46] Y. Han and Y. Shen, "Accurate spear phishing campaign attribution and early detection," in *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, 2016, pp. 2079–2086.

[47] V. Nguyen, "Attribution of spear phishing attacks: A literature survey," DEFENCE SCIENCE and TECHNOLOGY ORGANISATION EDINBURGH (AUSTRALIA), Tech. Rep., 2013.

[48] M. Khonji, Y. Iraqi, and A. Jones, "Phishing detection: A literature survey," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 4, pp. 2091–2121, 2013.

[49] S. Abu-Nimeh, D. Nappa, X. Wang, and S. Nair, "A comparison of machine learning techniques for phishing detection," in *Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit*, 2007, pp. 60–69.

[50] V. Zeng, S. Baki, A. E. Aassal, R. Verma, L. F. T. De Moraes, and A. Das, "Diverse datasets and a customizable benchmarking framework for phishing," in *Proceedings of the Sixth International Workshop on Security and Privacy Analytics*, 2020, pp. 35–41.

[51] K. Chiew, H. C. Ee, L. T. Choon, B. Johari, and K. C. Y. Sheng, "Building standard offline anti-phishing dataset for benchmarking," 2018.

[52] S. Bell and P. Komisarczuk, "An analysis of phishing blacklists: Google safe browsing, openphish, and phishtank," New York, NY, USA: Association for Computing Machinery, 2020, ISBN: 9781450376976.

[53] Akamai, "Phishing: Holiday season attacks on the rise," `https://blogs.akamai.com/2021/02/phishing-holiday-season-attacks-on-the-rise.html`, 2021.

[54] D. cyber grand challenge binaries, "Darpa cgc," in *https://github.com/CyberGrandChallenge*, 2016.

[55] A. Hazimeh, A. Herrera, and M. Payer, "Magma: A ground-truth fuzzing benchmark," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 4, no. 3, pp. 1–29, 2020.

[56] B. Dolan-Gavitt *et al.*, "Lava: Large-scale automated vulnerability addition," in *2016 IEEE Symposium on Security and Privacy (SP)*, 2016, pp. 110–121.

[57] UCI, "Phishing website dataset," `https://archive.ics.uci.edu/ml/datasets/phishing+websites`, 2015.

[58] D. G. Dobolyi and A. Abbasi, "Phishmonger: A free and open source public archive of real-world phishing websites," in *2016 IEEE conference on intelligence and security informatics (ISI)*, IEEE, 2016, pp. 31–36.

[59] C. L. Tan, "Phishing dataset for machine learning: Feature evaluation," *Mendeley Data*, vol. 1, p. 2018, 2018.

[60] R. S. Rao and A. R. Pais, "Two level filtering mechanism to detect phishing sites using lightweight visual similarity approach," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 9, pp. 3853–3872, 2020.

[61] O. Katz, "Working from home – the new threat frontier," in *https://blogs.akamai.com/2020/05/working-from-home-the-new-threat-frontier.html*, Akamai, 2021.

[62] D. K. T. Sec), "Social engineering toolkit," `https://github.com/trustedsec/social-engineer-toolkit`, 2020.

[63] X. Roche, "Httrack website copier," https://www.httrack.com/page/7/en/index.html, 2020.

[64] B. Wardman, T. Stallings, G. Warner, and A. Skjellum, "High-performance content-based phishing attack detection," in *eCrime Researchers Summit, eCrime*, 2011, ISBN: 9781457713392.

[65] A. Caliskan-Islam *et al.*, "De-anonymizing programmers via code stylometry," in *24th USENIX Security Symposium (USENIX Security 15)*, 2015, pp. 255–270.

[66] R. G. Aylin Caliskan, "De-anonymizing programmers from source code," DEFCON26, 2018.

[67] D. Roellke, A. Stein, E. Dauber, R. Harang, A. Caliskan, and R. Greenstadt, "Poster: Stylometry of author-specific and country-specific style features in javascript," ser. NDSS '18, 2018.

[68] A. Hidayat, "Ecmascript parsing infrastructure for multipurpose analysis," https://esprima.org/.

[69] N. Shuyo, "Langdetect 1.0.8," https://pypi.org/project/langdetect/, 2020.

[70] B. B. A. (BBA), "Cybercrime top 10 countries where attacks originate," https://www.bba.org.uk//wp-content/uploads/2015/02/red24CybercrimeTop10countrieswh 2015.pdf, 2020.

[71] P. de la Riva (buguroo), "Analyzing the world's top 3 cybercrime countries," https://www.buguroo.com/en/blog/the-worlds-top-3-cybercrime-and-online-fraud-hotspots, 2020.

[72] W3Techs, "Historical yearly trends in the usage statistics of content languages for websites," 2020.

[73] E. Union, "What is considered personal data under the eu gdpr?," https://gdpr.eu/eu-gdpr-personal-data/, 2018.

[74] S. of California Department of Justice, "California consumer privacy act (ccpa)," https://oag.ca.gov/privacy/ccpa, 2018.

[75] 98th United States Congress, "Computer fraud and abuse act," 1986.

[76] C. of Federal Regulations, "2 cfr § 200.79 - personally identifiable information (pii)," https://www.law.cornell.edu/cfr/text/2/200.79, 2014.

[77] G. C. A. Center, "Cloud data loss prevention - fully managed service designed to help you discover, classify, and protect your most sensitive data.," https://cloud.google.com/dlp, 2020.

[78] U. D. of Labor, "Guidance on the protection of personal identifiable information," https://www.dol.gov/, 2010.

[79] ——, "Health insurance portability and accountability act of 1996," https://aspe.hhs.gov/report/health-insurance-portability-and-accountability-act-1996, 1996.

[80] S. of California Department of Justice, "The california privacy rights act of 2020," https://oag.ca.gov/, 2020.

[81] 108th United States Congress, "The controlling the assault of non-solicited pornography and marketing act," 2003.

[82] 106th United States Congress, "Gramm-leach-bliley act," 1999.

[83] 102nd United States Congress, "The telephone consumer protection act of 1991," 1991.

[84] 91st United States Congress, "The fair credit reporting act," 1970.

[85] 99th United States Congress, "The electronic communications privacy act," 1986.

[86] G. C. A. Center, "What is considered personal data under the eu gdpr?," https://cloud.google.com/architecture/de-identification-re-identification-pii-using-cloud-dlp, 2020.

[87] ——, "What is considered personal data under the eu gdpr?," https://cloud.google.com/dlp/docs/infotypes-reference, 2020.

[88] S. Tzu, "The art of war," 1772.

[89] R. Law, "The sovereign internet bill," 2015.

[90] C. A. of China, "The golden shield project," 2013.

[91] "The dark side of russia: How new internet laws and nationalism fuel russian cybercrime," insights.com, 2020.

[92] A. Greenberg, "The biggest dark web takedown yet sends black markets reeling," in *Wired*, https://www.wired.com/story/alphabay-takedown-dark-web-chaos, 2017.

[93] S. L. Joshua Davis, "The untold story of silk road," in *Wired*, https://www.wired.com/2015/04/silk-road-1/, 2015.

[94] T. N. Ed Caesar, "The cold war bunker that became home to a dark-web empire," https://www.newyorker.com/magazine/2020/08/03/the-cold-war-bunker-that-became-home-to-a-dark-web-empire, 2020.

[95] X. Han, 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*, ser. CCS '16, Vienna, Austria: Association for Computing Machinery, 2016, 1402–1413, ISBN: 9781450341394.

[96] "Discovering organizational hierarchy through a corporate ranking algorithm: The enron case." complexity," to be published, 2022.

[97] *Fred Cohen & Associates*, [Online; accessed 1. Nov. 2021], 2016.

[98] N. Provos *et al.*, "A virtual honeypot framework.," in *USENIX Security Symposium*, vol. 173, 2004, pp. 1–14.

[99] B. M. Bowen, P. Prabhu, V. P. Kemerlis, S. Sidiroglou, A. D. Keromytis, and S. J. Stolfo, "Botswindler: Tamper resistant injection of believable decoys in vm-based hosts for crimeware detection," *RAID*, 2010.

[100] B. M. Bowen, S. Hershkop, A. D. Keromytis, and S. J. Stolfo, "Baiting inside attackers using decoy documents," *Security and Privacy in Communication Networks (SecureComm)*, 2009.

[101] Y. Park and S. J. Stolfo, "Software decoys for insider threat *," *AsiaCCS*, 2012.

[102] X. Han, N. Kheir, and D. Balzarotti, "Phisheye: Live monitoring of sandboxed phishing kits," *CCS*, 2016.

[103] S. Zawoad, A. K. Dutta, A. Sprague, R. Hasan, J. Britt, and G. Warner, "Phish-net: Investigating phish clusters using drop email addresses," IEEE Computer Society, 2013.

[104] A. Oest *et al.*, "Phishtime: Continuous longitudinal measurement of the effectiveness of anti-phishing blacklists phishtime: Continuous longitudinal measurement of the eeectiveness of anti-phishing blacklists," *Usenix*, 2020.

[105] X. Han, N. Kheir, and D. Balzarotti, "Deception techniques in computer security: A research perspective," *ACM Comput. Surv*, vol. 51, 2018.

[106] A. Aleroud and L. Zhou, *Phishing environments, techniques, and countermeasures: A survey*, Jul. 2017.

[107] V. Nguyen, "Attribution of spear phishing attacks: A literature survey,"

[108] *Humboldt: A Distributed Phishing Disruption System*. IEEE, 2009, ISBN: 9781424446261.

[109] J. Gustafson and J. Li, "Leveraging the crowds to disrupt phishing," IEEE Computer Society, 2013, pp. 82–90, ISBN: 9781479908950.

[110] C. Yue, H. Wang, H Wang, and C Yue, "Bogusbiter: A transparent protection against phishing attacks," *ACM Transactions on Internet Technology*, vol. 10, 6 2010.

[111] M. Akiyama, T. Yagi, T. Hariu, Y. Kadobayashi, and T. Yagi, "Honeycirculator: Distributing credential honeytoken for introspection of web-based attack cycle," *International Journal of Information Security*, vol. 17, pp. 135–151, 2018.

[112] P. Snyder, L. Ansari, C. Taylor, and C. Kanich, "Browser feature usage on the modern web," vol. 14-16-November-2016, Association for Computing Machinery, Nov. 2016, pp. 97–110, ISBN: 9781450345262.

[113] F. Monrose and A. Rubin, "Authentication via keystroke dynamics," in *Proceedings of the 4th ACM Conference on Computer and Communications Security*, ser. CCS '97, Zurich, Switzerland: Association for Computing Machinery, 1997, 48–56, ISBN: 0897919122.

[114] ——, "Authentication via keystroke dynamics," 1997.

[115] M. Andreolini, A. Bulgarelli, M. Colajanni, and F. Mazzoni, "Honeyspam: Honeypots fighting spam at the source," p. 77.

[116] J. Deblasio, S. Savage, G. M. Voelker, and A. C. Snoeren, "Tripwire: Inferring internet site compromise," vol. 14, 2017.

[117] A. Pitsillidis *et al.*, "Botnet judo: Fighting spam with itself."

Appendix A: PII Reference List

PII Category	Indicator	Google Cloud DLP
USRNAME	Username/	-
USRNAME	UserID/	-
USRNAME	Signon/	-
USRNAME	Login/	-
USRNAME	AppleID/	-
USRNAME	AdobeID/	-
USRNAME	AOLuser/	-
USRNAME	Yahoouser/	-
USRNAME	Hotmailuser/	-
USRNAME	Gmailuser/	-
PSWD	Password/	PASSWORD
PSWD	Passcode	PASSWORD
PSWD	Security Question/Mother's maiden name	PASSWORD
PSWD	Security Question	PASSWORD
PSWD	Shared Secret	PASSWORD
PSWD	Recover	PASSWORD
PSWD		PASSWORD
AIRACCT	Airline Miles Number/	-
AIRACCT	AA Advantage Number/	-
AIRACCT	Mileage Plus/	-
AIRACCT	True Blue/	-
AIRACCT	Sky Miles	-
EDUCATION	School identification/	-
EDUCATION	Highest Degree Earned	-
ORIGIN	Alienage-Citizenship	-
AUTOVIN	License Plate Numbers/	VEHICLE_ID
AUTOVIN	Tag number	VEHICLE_ID
TOLL	EZpass number	-
AUTOVIN	Auto VIN number	VEHICLE_ID
INSURANCE	Insurance Policy Number	ICD9_CODE
INSURANCE	Medical Insurance Number	ICD9_CODE
INSURANCE	Group No.	ICD10_CODE
INSURANCE	Medicaid Number	ICD10_CODE
INSURANCE	Medicare Number	ICD10_CODE
MEDICAL	Caregiver Status	MEDICAL_TERM
BIOMETRIC	Color	-
CREDIT	Credit History	-
ORIGIN	Creed	-
MEDICAL	Disability	-
FAMSTAT	Familial Status	-
GENDER	Gender	GENDER
GENDER	Gender Identity	GENDER
MEDICAL	Lactation accommodation	
FAMSTAT	Marital status	-

ORIGIN	National Origin	-
MEDICAL	Pregnancy	MEDICAL_TERM
ORIGIN	Race	ETHNIC_GROUP
ORIGIN	Religion	-
EMPLOYMENT	Employer information/	-
EMPLOYMENT	Salary History	-
GENDER	Sex	GENDER
MEDICAL	Sexual health	MEDICAL_TERM
PREFERENCE	Sexual orientation	-
EMPLOYMENT	Unemployment status	-
EMPLOYMENT	Veteran or active military status	-
NAME	Name/	FEMALE_NAME
NAME	First/	FIRST_NAME
NAME	Middle/	MALE_NAME
NAME	Middle Initial/	MALE_NAME
NAME	Last	LAST_NAME
AGE	Age/Birthdate/	AGE
AGE	Birth month/Birth Day/Birth Year	DATE
ADDRESS	Address/	EMAIL_ADDRESS
ADDRESS	House Address/	LOCATION
ADDRESS	Country/	LOCATION
ADDRESS	State/	LOCATION
ADDRESS	Postal Code/ZIP Code	LOCATION
ADDRESS	Street/Apartment	STREET_ADDRESS
EMAIL	Email/	EMAIL_ADDRESS
EMAIL	Contact email	EMAIL_ADDRESS
PHONE	Phone/ Fon	PHONE_NUMBER
PHONE	Mobile	PHONE_NUMBER
FINANCIAL	Bank Account Number/	by country
CREDIT	Credit Card/	CREDIT_CARD_NUMBER
CREDIT	cc	CREDIT_CARD_TRACK_NUMBER
CREDIT	Debit Card/	-
CREDIT	CVV/	-
CREDIT	Expiration Date/	-
FINANCIAL	Checking Account/	by country
FINANCIAL	Routing Number/	US_BANK_ROUTING_MICR
BROKERAGE	Bitcoin Account	-
BROKERAGE	Wallet	-
BROKERAGE	Brokerage Account/	-
PASSWORD	PIN	PASSWORD
SOCIALSEC	Social security number/	US_SOCIAL_SECURITY_NUMBER
SOCIALSEC	Last four digits	US_SOCIAL_SECURITY_NUMBER
TRAVEL	Global Entry number/	-
TRAVEL	Redress Number	-
PASSPORT	Passport Number	PASSPORT
DRIVE	Drivers License/	DRIVERS_LICENSE
DRIVE	Issuing State	DRIVERS_LICENSE
LEGAL	Arrest/	DOCUMENT_TYPE/LEGAL/*
LEGAL	Conviction Record	DOCUMENT_TYPE/LEGAL/*
GENETIC	Genetic predisposition – or carrier status	-
BIOMETRIC	Biometric information	-
LEGAL	Victim (violence, stalking, ...)	DOCUMENT_TYPE/LEGAL/*
FINANCIAL	TAN	-
FINANCIAL	SWIFT	SWIFT_CODE

FINANCIAL	BIC	SWIFT_CODE
FINANCIAL	IBAN	IBAN_CODE
FINANCIAL	Transaction Number	IBAN_CODE
FINANCIAL	Transaction ID	IBAN_CODE

Table A.1: A comparison of our newly proposed reference list to Google Cloud DLP. We show 36 categories, indicated by 356 identifiers, whereas Google Cloud DLP has only 28 categories.

Appendix B: PII Category Distribution

Relative proportions of PII categories per year.

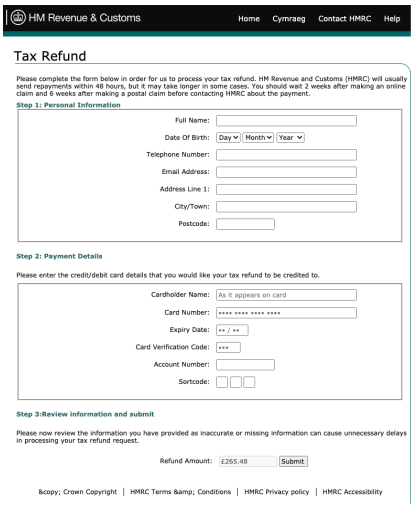
	2016	2017	2018	2019
email	16.57%	14.14%	13.42%	9.45%
contact	13.00%	11.57%	11.63%	9.97%
location	3.64%	3.98%	4.14%	4.26%
name	10.33%	14.75%	10.01%	13.27%
age	1.88%	1.47%	2.13%	2.32%
education	1.30%	1.59%	1.45%	2.80%
birthday	0.00%	0.06%	0.11%	0.04%
work	1.04%	0.55%	0.84%	1.50%
subscription	0.06%	0.92%	0.00%	0.24%
username	22.42%	15.85%	20.25%	19.09%
password	13.91%	11.69%	11.19%	9.47%
insurance	0.52%	0.43%	0.22%	0.66%
auto	0.00%	0.06%	0.00%	0.11%
citizenship	0.32%	0.31%	0.06%	0.26%
unemployment	0.00%	0.00%	0.00%	0.00%
orientation	0.13%	0.00%	0.22%	0.12%
military	0.00%	0.00%	0.06%	0.01%
religion	0.00%	0.00%	0.00%	0.00%
physical	0.26%	0.43%	0.06%	0.40%
airline	1.43%	0.55%	1.06%	1.13%
medical	0.26%	0.12%	0.11%	0.31%
toll	0.00%	0.00%	0.06%	0.07%
questions	0.26%	0.18%	0.34%	0.38%
financial	5.33%	9.12%	10.85%	10.83%
travel	0.06%	0.06%	0.34%	0.18%
tax	4.81%	8.63%	8.05%	10.36%
driving	1.23%	2.39%	1.34%	1.30%
legal	0.45%	0.61%	0.56%	0.43%
biometric	0.45%	0.12%	0.56%	0.31%
wire	0.32%	0.43%	0.95%	0.73%

Appendix C: Natural Language Distribution

Relative increase or decrease per language when comparing phishing to the the languages count in the benign part of the internet according to [72].

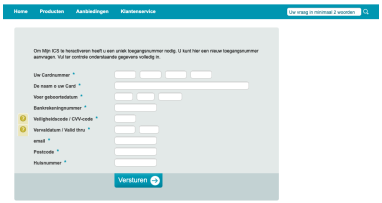
	2016	2017	2018	2019
English	0.1009	0.09	0.1263	-5.02%
German	-0.07%	-1.08%	0.0036	-0.71%
French	0.0192	-1.33%	0.0101	0.0191
Portuguese	-0.79%	0.0258	-0.37%	0.0193
Spanish	-3.75%	-2.49%	-4.04%	-1.20%
Italian	0.0048	-0.65%	-1.12%	0.0003
Russian	-4.86%	-5.93%	-5.52%	-2.79%
Dutch	0.0003	-0.22%	-0.34%	0.004
Catalan	0.0047	0.0265	0.0043	0.0117
Romanian	0.0065	0.0141	0.0034	0.0106
Indonesian	0.0103	0.0021	0.0109	0.0039
Afrikaans	-	-	-	-
Danish	0.0056	0.0103	0.0066	0.0076
Norwegian	0.0076	0.0178	0.0043	0.0064
Welsh	-	-	-	-
Swedish	0.0026	0.0068	0.0003	-0.03%
Polish	-1.52%	-1.23%	-0.96%	-0.49%
Vietnamese	0.0007	0.0003	-0.39%	0.0017
Korean	-0.32%	-0.59%	-0.57%	0.0021
Bulgarian	-0.01%	-0.20%	0.0065	0.0087
Japanese	-4.33%	-5.70%	-5.50%	-2.04%
Estonian	0.0009	0.0021	0.0043	0.0063
Turkish	-1.31%	-1.52%	-0.87%	-0.44%
Slovenian	0.00%	0.0068	0.0011	0.0042
Swahili	-	-	-	-
Macedonian	-	-	-	-
Slovak	0.0008	-0.14%	0.0013	-0.02%
Arabic	-0.80%	-0.41%	-0.38%	0.0001
Croatian	0.0019	0.0021	0.0022	0.0013
Ukrainian	-0.10%	-0.10%	0.0044	0.0029
Albanian	-	-	-	-
Finnish	-0.01%	0.0001	-0.09%	0.0002
Hungarian	-0.21%	0.0007	-0.50%	-0.31%
Kannada	-	-	-	-
Czech	-0.80%	-0.49%	-0.58%	-0.80%
Latvian	0.0009	0.0021	0.0001	-0.02%
Lithuanian	0.0009	0.0014	-0.20%	0.0004
Greek	-	-	-	-
Hebrew	-0.20%	-0.20%	0.0012	–0.0%
Thai	-0.30%	-0.30%	-0.09%	–0.0%
Persian	-1.10%	-1.42%	-1.70%	-1.68%
Hindi	-	-0.10%	-0.10%	-0.06%

Appendix D: Sample PII Phishing Sites



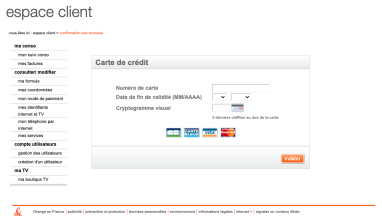
(a) A phishing site of extreme danger to PII. It seems very authentic and believable when compared to many other phishing attempts.

(b) A phishing site of extreme danger to PII. It seems very authentic and believable when compared to many other phishing attempts.

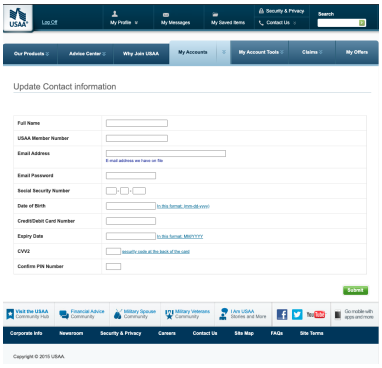


(c) A phishing site of extreme danger to PII. It seems very authentic and believable when compared to many other phishing attempts.

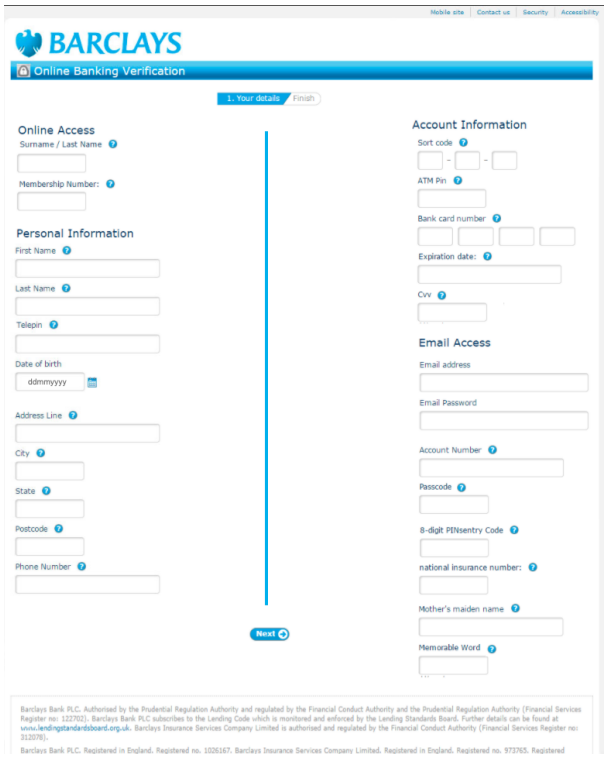
(d) A phishing site of extreme danger to PII. It seems very authentic and believable when compared to many other phishing attempts.



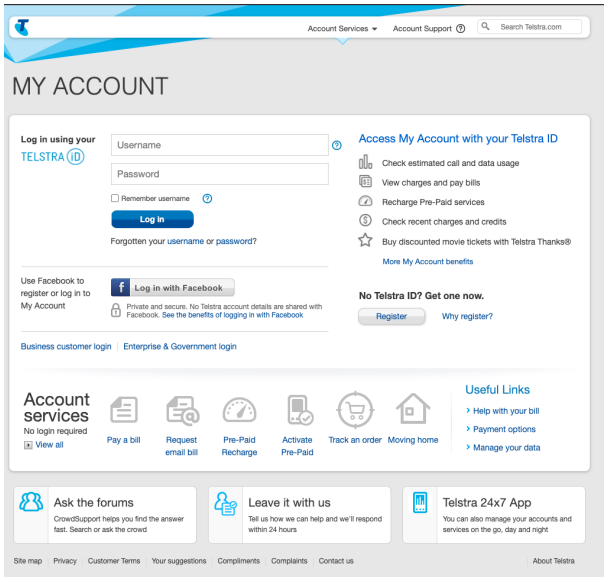
(e) A phishing site of extreme danger to PII. It seems very authentic and believable when compared to many other phishing attempts.



(f) A phishing site of extreme danger to PII. It seems very authentic and believable when compared to many other phishing attempts.



(g) A phishing site of extreme danger to PII. It seems very authentic and believable when compared to many other phishing attempts.



(h) A phishing site of extreme danger to PII. It seems very authentic and believable when compared to many other phishing attempts.

Figure D.1: A benign site and its malicious clone, utilizing identical variable names - an indicator that can be used to link a site to its clonee

Appendix E: URLs

ID	Brand	Autfill	Email & Passwd	Credit Card	Bank Account	Address	Phone Number	SSN	3D-Secure	File Upload
97651403	ERR_ADDRESS_UNREACHABLE									
97651402	DNS_PROBE_FINISHED_NXDOMAIN									
97650291	REDIRECT									
97649635	The QR Code has been paused.									
97646231	NET::ERR_CERT_COMMON_NAME_INVALID 404									
97646149	DNS_PROBE_FINISHED_NXDOMAIN									
97645729	404									
97635036	DNS_PROBE_FINISHED_NXDOMAIN									
97634911	DNS_PROBE_FINISHED_NXDOMAIN									
97634910	DNS_PROBE_FINISHED_NXDOMAIN									
97652961	PayPal							x	x	x
97627949	PayPal									
97627948	PayPal	artifact of an anti-phishing workshop								
97624840	ERR_BLOCKED_BY_CLIENT REDIRECT TO BENIGN (cloaked?)									
97624837	ERR_BLOCKED_BY_CLIENT									
97624179	ERR_BLOCKED_BY_CLIENT									
97624177	ERR_BLOCKED_BY_CLIENT									
97653094	404									
97653093	404									
97653087	ERR_CONNECTION_TIMED_OUT									
97653071	PayPal							x	x	
97653070	ERR_CONNECTION_TIMED_OUT									
97653068	ERR_CONNECTION_TIMED_OUT									
97653064	PayPal	fraudulent money request								
97655960	ERR_CONNECTION_TIMED_OUT									
97655730	PayPal					x	x			
97655729	PayPal Spanish		x							
97667703	403 Forbidden									
97667691	PayPal		x	x						
97667687	403 Forbidden									
97667401	File Not Found									
97667034	File Not Found									
97666966	French		x				x			
97666963	File Not Found									
97666954	French		x							
97666858	REDIRECT									
97665891	ERR_CONNECTION_REFUSED									
97666309	DNS_PROBE_FINISHED_NXDOMAIN									
97665191	Blank page									

ID	Brand	Autfill	Email & Passwd	Credit Card	Bank Account	Address	Phone Number	SSN	3D-Secure	File Upload
97671610	CLOAKED		x	x	x	x				x
97670264	Hoster takedown									
97669569	DNS_PROBE_FINISHED_NXDOMAIN									
97669558	404									
97669383	Hoster takedown									
97669373	Hoster takedown									
97669268	ERR_CONNECTION_TIMED_OUT									
97668872	403									
97668424	PayPal		x	x	x	x				x
97668329	503									
97668331	503									
97668328	503									
97667862	Blank page									
97667861	Not Found									
97687022	DNS_PROBE_FINISHED_NXDOMAIN									
97687021	DNS_PROBE_FINISHED_NXDOMAIN									
97686431	DUPLICATE									
97686430	PayPal (mobile agent)		x							
97686372	404									
97686361	File not found.									
97684512	PayPal									
97704766	PayPal	log in button dead?								
97704530	PayPal	does not allow . in email								
97692475	PayPal		x	x						
97691363	302									
97691360	302									
97691362	302									
97691361	302									
97691359	302									
97691349	302									
97691350	302									
97691348	302									
97691351	302									
97691179	302									
97691178	302									
97701751	PayPal	disabled pasting credit card credit card exp. date max. 2026								
97710858	Forbidden									
97690566	404									
97690122	Internal Server Error									
97754193	REDIRECT									
97753391	hoster take down									
97749501	PayPal	login just fails, backend dead?	x							
97746649	PayPal	mobile only, fake customer chat								
97746171	ERR_BLOCKED_BY_CLIENT									
97746163	PayPal	just asking MFA code w/o asking phone first								
97741308	paypal		x	x	x	x			x	x
97725530	paypal		x	x	x	x				
97725477	paypal	stalls during sms retrieval doesnt verify password at all								
97743898	ERR_CONNECTION_REFUSED									

Table E.1