Beneath the Phishing Scripts: A Script-Level Analysis of Phishing Kits and Their Impact on Real-World Phishing Websites

Woonghee Lee
Korea University
Seoul, Republic of Korea
whlee@isslab.korea.ac.kr

Junbeom Hur
Korea University
Seoul, Republic of Korea
jbhur@korea.ac.kr

Doowon Kim
University of Tennessee
Knoxville, Tennessee, USA
doowon@utk.edu

ABSTRACT

Phishing kits have become increasingly popular among cybercriminals because they offer an easy-to-use and efficient way for phishing attackers to build phishing websites. Prior work on phishing kits has focused on analyzing specific behavioral features (e.g., evasion techniques), and measuring their effectiveness on the anti-phishing mechanisms. Unfortunately, such prior studies provide a limited perspective, either targeting specific phishing kits or not fully addressing the server-side strategies at the script level that offer insights into the phishing attacker's view.

In this paper, we systematically conduct a comprehensive study of phishing kits at the script level, aiming to better understand the server-side behavior. Particularly, we design a crawler that periodically (every 15 mins) collects phishing kits used in the wild and client-side resources of real-world phishing websites (e.g., index.html, images, CSS, JavaScript) for 18 months. We utilize two types of our collected dataset (4,153 phishing kits and 2.4M phishing webpages) for our study. First, we classify user interaction patterns based on information-exfiltrating components of phishing kits into three categories: single-stage (non-real-time) phishing, multi-stage (non-real-time) phishing, and multi-stage (real-time) phishing. We then identify the potential information leakage of each pattern. Next, we conduct an in-depth script-level analysis of the evasive behaviors in the kits. Aiming to evaluate their practical impact on the phishing sites, we also measure how many phishing kits are used and deployed with an emphasis on the chronological trends for phishing attacks by clustering the landing pages of phishing kits with those of our collected phishing websites. Lastly, we discuss the security implications of our comprehensive study on web interaction, URL patterns/redirection, and kit-deployment trends in the phishing detection literature.

CCS CONCEPTS

• **Security and privacy → Web application security**.

KEYWORDS

Phishing; Phishing Kit; User Interaction; Evasion Technique;

ACM Reference Format:
Woonghee Lee, Junbeom Hur, and Doowon Kim. 2024. Beneath the Phishing Scripts: A Script-Level Analysis of Phishing Kits and Their Impact on Real-World Phishing Websites. In *ACM Asia Conference on Computer and Communications Security (ASIA CCS '24), July 1–5, 2024, Singapore, Singapore.* ACM, New York, NY, USA, 17 pages. https://doi.org/10.1145/3634737.3657013

1 INTRODUCTION

Phishing is a type of social engineering attack where attackers set up a fraudulent website designed to look like a legitimate website and to lure users into putting their personal information. Such social engineering attacks currently affect billions of Internet users [38, 68]. Even worse, phishing kits facilitate the easy construction and deployment of phishing websites for even novice phishing attackers without in-depth knowledge of web development, such as PHP and JavaScript. A phishing kit typically consists of landing pages (*i.e.*, fraudulent webpages that mimic target brands), exfiltration methods (*e.g.*, email, Telegram, and administrator panel), and evasion techniques against anti-phishing techniques [53, 70].

Phishing kits have gained much attention recently. For example, studies on access blockers [51, 53] and client-side evasion techniques [70, 71] pointed out the limitations of anti-phishing mechanisms (*e.g.*, Google Safe Browsing) when blocking phishing attacks. The other recent studies on the lifecycle of phishing kits [28, 36] and leaked phishing data [54, 62, 65] examined the damages and duration of phishing campaigns.

However, there are many unveiled characteristics of phishing kits that should be better understood to mitigate phishing attacks. *First*, prior studies are confined to focusing only specific phishing kits, such as financial brands within a specific country [28, 54]. Similarly, research on two-factor authentication (2FA) bypassing phishing has exclusively examined three man-in-the-middle (MITM) phishing kits [41]. *Second*, server-side strategies have not been fully analyzed. For example, one recent study on web interaction of phishing sites [61] only checked if phishing sites have multiple pages, having a limitation in figuring out the precise processes and amount of data exposure caused by a phishing attack. We observed this limitation is caused by the absence of an in-depth analysis at the script level of phishing kits, which can effectively capture server-side behavior, such as real-time interaction between the phishing attacker and the phishing site. To this end, our work aims to systematically explore the behavioral strategies of phishing kits through script-level analysis, analyze how these strategies manifest on real phishing websites,

ASIA CCS '24, July 1–5, 2024, Singapore, Singapore

ACM ISBN 979-8-4007-0482-6/24/07…$15.00
https://doi.org/10.1145/3634737.3657013

and discuss how our findings uncovered during the analysis can contribute to phishing detection techniques.

Specifically, we first design a crawler that periodically (*i.e.*, every 15 mins) accesses real-world phishing websites in the wild and collects two datasets from the phishing websites: (1) client-side resources of phishing websites (*e.g.*, index.html, JavaScript, images, CSS, etc.) and (2) phishing kits. The crawler is fed from one of the largest phishing URL repositories, APWG eCX (Anti-Phishing Working Group eCrime Exchange) [9]. For 18 months (Aug. 2021 – Jan. 2023), the crawler successfully collects distinct 4,153 phishing kits and 5,178,919 (5.2M) phishing webpages (*i.e.*, client-side resources) from 14,373,422 (14.4M) real-world phishing URLs. We analyze the collected datasets at the script level to establish the relationship between the intricate details of phishing kits and real-world phishing ecosystem. Particularly, we *first* systematically classify user interaction patterns — *single stage (non-real-time)*, *multi-stage (non-real-time)*, and *multi-stage (real-time) phishing* — and analyze their characteristics based on how the kits handle and leak the information submitted by victims. *Second*, we comprehensively investigate the two underexplored evasive behavioral features of phishing kits — redirection and dynamically-generation of URLs — and measure their real-world implications. *At last*, we find out how many phishing kits are used and deployed to launch phishing attacks. Each phishing website has a unique fingerprint on its landing pages, even though they are visually identical. To address this, we collect landing page scripts by deploying phishing kits on local servers. Then, we score the structural similarities between the scripts extracted from phishing kits and those from our collected phishing webpages dataset and cluster the landing pages using the structural similarity scores.

With forementioned methodologies, we unveiled the following hidden characteristics of phishing kits. *First*, each classified type of phishing kit significantly differs in the submitted information type, the number of interactions, targeting brands, and exfiltrating methods. *Second*, our script-level analysis observes that 60% of kits actively use redirection and 1.5M phishing URLs adopt the redirection feature of the kits as an evasion technique. We identify six common patterns of dynamically-generated URLs and disclose that the three most common patterns — random names of directories, incremental numbers, and random HTTP GET values — account for over 90% of dynamically-generated URLs in the wild. Furthermore, we discuss the differences between the URLs generated by phishing kits and those of actual web services targeted by phishing. *Lastly*, comparing clustered landing page scripts from phishing kits to HTML scripts from real-world phishing sites, we find that a single kit is deployed on an average of 46.92 domains, showing the predominant usage of a specific kit in the phishing ecosystem. Moreover, we measure the monthly kit-deployment trend and find that 107 kits, most of which target banking brands, show the bursty deployment trend. Additionally, we discuss how our findings based on web interaction, URL patterns & redirection, and kit-deployment trends can contribute to phishing detection.

Our work has the following contributions.

- We conduct a comprehensive study to systematically disclose the correlations between phishing kits and websites in the wild.
- We classify user interaction patterns of phishing kits into three types and analyze their characteristics in terms of information leakage flow, including interaction trials, information exfiltration types, and methods.
- We comprehensively analyze the behavioral features (*e.g.*, redirection and dynamically-generation of URLs) of phishing kits at the script level and measure how such features affect phishing attacks in the real world.
- We utilize and compare landing pages of phishing kits to real-world phishing pages to better understand how phishing kits are used and deployed for phishing attacks.
- We discuss the security implications to give insights into building more efficient phishing detection based on our findings about web interaction, URL patterns & redirection, and kit-deployment trends.

2 BACKGROUND

In this section, we provide a brief overview of phishing attacks and phishing components in the kits.

2.1 Phishing Attacks

Phishing attacks are typically composed of three steps. *First*, phishing attackers build a deceptive website that disguises itself as a legitimate one, such as paypal.com, apple.com, or google.com. *Second*, as the main purpose of phishing attacks is to obtain victims' credentials, the attackers send phishing messages to potential victims via diverse channels, such as emails [9, 60, 64], text messages and phone calls [40, 66], or direct messages (DM) over social media [27, 31]. For example, the attackers may send an email requiring urgent action from victims, such as "Your email address has been compromised. You need to reset your password right now." *Lastly*, the attackers exfiltrate the victims' credentials from the phishing websites after victims are tricked into entering their credentials into the phishing websites.

2.2 Phishing Kits

The phishing kit is defined as a 'ready-to-deploy' package that helps attackers readily deploy phishing websites [32]. These kits typically consist of three general components handling 1) website appearance, 2) victims' information (*e.g.*, credentials), and 3) evasion techniques.

Website Appearance Components. It consists of phishing website code (*e.g.*, HTML, JavaScript, CSS) and images (*e.g.*, brand logo) of the target brand that look similar to the original website, luring victims to submit their credentials. However, prior works [25, 30] have shown that some phishing pages do not contain exactly the same visual content as the original websites; rather, they show various webpages with different layouts and contents mimicking the same brand.

Information Exfiltrating Components. This component handles victims' information (*e.g.*, credentials) and exfiltrates the information for the attacker using various ways such as email and Telegram [47, 63]. Phishing kit users are able to send emails containing victims' information by utilizing the PHPmailer [3] library included in phishing kits or the built-in PHP mail function. When using Telegram as an exfiltration method, for example, phishing kit users send victims' credential information by specifying their chatbot IDs and tokens in scripts using cURL [1]. More advanced kits may have an administrator panel that facilitates victims' information

management for phishing attackers [28]. For example, the `uAdmin` phishing kit family provides various exfiltration methods; one allows real-time communication between the administrator panel and the phishing webpage. This helps phishing attackers defeat the two-factor authentication defense mechanisms by acting as a man-in-the-middle through real-time interaction with the victims [41].

Evasion Technique Components. Advanced kits also have evasion technique components that help phishing attackers bypass anti-phishing systems, such as Google Safe Browsing [14]. The evasion techniques can be categorized into two approaches: server-side and client-side [26, 51–53, 70, 71]. For client-side ones, kits provide JavaScript scripts that show Captcha pages or detect users' mouse movements and clicks [70, 71], in order to identify if the visitors are real humans. For server-side ones, there are four typical evasion techniques [43, 47, 51, 53] as follows.

- *.htaccess.* This file is generally used for Apache web servers to block or redirect access from clients. In phishing kits, it is particularly used to block access from potential anti-phishing agents by specifying IPs, hostnames, referring URLs, and user agents (*i.e.*, bots).
- *robots.txt.* This is a text file that is used to allow or block web crawlers and bots [42]. For example, 'User-agent: * Disallow: /' blocks all web crawlers from all content. However, this file does not have any compulsory restrictions for crawlers and bots, implying that bots (including anti-phishing bots) may not follow the rules and can access phishing websites. In the phishing ecosystem, in spite of the lack of its compulsion, phishing websites keep using the files because the files help evade the crawling by bots of the major search engines, making their phishing site URLs less noticeable.
- *Blocker.* A blocker is a special-purpose PHP script inside kits. It blocks requests from anti-phishing bots similar to *.htaccess* and `robots.txt` files. It is more flexible for phishing attackers to configure the blocking rules than *.htaccess*. This is because phishing kits with blockers can be used on any web server, while *.htaccess* can be used only for Apache web servers.
- *Dynamically-Generated URLs.* Phishing websites dynamically generate phishing URLs for each victim by creating new directories, scripts, or HTTP GET values [47]. In other words, every victim has different phishing URLs. Since anti-phishing systems such as blocklists normally block access to phishing websites based on URLs (not a domain) [51], the dynamically-generated directory can be effective in avoiding being added to a phishing blocklist, which helps extend the lifespan of phishing websites.

3 MOTIVATION & RESEARCH QUESTION

As numerous phishing kit families have kept emerging and evolving, the prior measurement and analysis studies mainly focus on certain confined features (or components) of phishing kits without systematic analysis at the script level. Due to the lack of it, the questions of how each component is implemented for each functionality of a phishing kit and how it is associated with or determines the user interaction pattern remain unanswered. Moreover, the behavioral connection between the user of visible phishing sites and the server (such as the information submission process) of phishing kits has been poorly understood.

Phishing Webpages	Number of Accessed URLs	14,373,422
	Number of Crawled URLs	2,483,870
	Number of Accessed Domains	844,718
	Number of Crawled Domains	497,856
Phishing Kits	Number of Crawled Archive Files	10,082
	Number of Distinct Phishing Kits	4,153
Kit Hit Rate per Domain		0.83%

Table 1: Phishing Kits and Webpages Collection.

One recent study [61] analyzed user interaction patterns of phishing sites in the real-world and identified multi-stage phishing patterns that trick potential victims into believing the site is legitimate and submitting their information. Unfortunately, due to the challenge of determining a relationship between phishing kits and phishing webpages, it is still *little* known which phishing kits have been used (*i.e.*, deployed) in the phishing ecosystem. This gap in knowledge highlights the need for a comprehensive study investigating the correlation between phishing kits and real-world phishing websites through script-level analysis. Such a study can provide valuable insights into constructing advanced phishing detection techniques by understanding phishing kits at a more fundamental level. To this end, in this paper, we seek to answer the following research questions:

- **RQ1.** How can user interaction patterns of the phishing kits be classified at the script level, and how does each type work with information-exfiltrating methods inside the kits?
- **RQ2.** How is each evasion-related behavior implemented in phishing kits at the script level, and how does it affect real-world phishing websites in the wild?
- **RQ3.** How many phishing websites share the same web appearance components of each kit, and how much each kit is used to deploy in the wild with what chronological trends?

4 DATASET COLLECTION

We describe how we design our crawler that periodically accesses phishing websites and collects their client-side web resources and phishing kits. Our phishing dataset is collected from Aug. 2021 to Jan. 2023 (18 months).

4.1 Phishing URLs & Client-side Resources

In order to obtain phishing URLs used in the real world, we utilize `eCrime` Exchange (eCX) [8], one of the largest phishing blocklist repositories operated by Anti-Phishing Working Group (APWG). APWG is an anti-phishing industry association and collects phishing URLs used for real phishing attacks in the wild from various sources, such as its industry members. APWG eCX has been widely used to analyze and better understand phishing ecosystems [35, 51–54, 62, 70, 71]. During the collection period from Aug. 22, 2021, to Jan. 31, 2023 (18 months), we have collected a total number of 14,373,422 (14.4M) phishing URLs from the eCX phishing URL repository. Table 1 summarizes our collected phishing dataset.

Our crawler is periodically (every 15 mins) fed with the phishing URLs from APWG eCX, and then visits the phishing URLs to collect phishing websites (including client-side web resources) and phishing kits. We visit each URL using a Selenium Chrome WebDriver [18] with `fake-useragent` [21] library in Python to

avoid the evasion technique of phishing kits. Out of a total number of 14.4M phishing URLs, the crawler is able to successfully visit 2,483,870 phishing URLs (17.28%). This is because many of them have become offline when our crawler accesses them because phishing websites typically have a very short life cycle [54]. An average of 27,274 phishing URLs are accessed every day.

Of the total 14.4M phishing URLs, 844,718 distinct domains are found; each domain, on average, has 17 different phishing URLs. Particularly, a certain domain[1] has more than 1.4M URLs, which is the largest number of phishing URLs reported from a single domain. This indicates that phishing domains are typically served with more than one URL, which would be facilitated by the dynamically-generated URL feature that helps evade the anti-phishing techniques – we will discuss the dynamically-generated URL feature more in Section 6.2.

Client-side Resources Collection. To answer our research question *RQ3*, our crawler collects the phishing websites' client-side web resources (HTML pages, JavaScript, CSS, images, screenshots, header information, etc.) by visiting the fed phishing URLs from APWG eCX. The crawler is implemented with Selenium WebDriver [18], which can help simulate real users' access to phishing websites as it fully loads and executes all client-side resources, such as JavaScript, CSS, and images on the webpages. First, we filter out the webpages showing phishing blocking pages of hosting services and HTTP error response codes, such as '404 not found' or '403 forbidden.' This is because we attempt to obtain the real landing page scripts for the URL requested. Finally, we could successfully crawl 2,483,870 landing page scripts from 14,373,422 phishing URLs accessed. These scrapped phishing webpages will be used to analyze the internal redirection flows in the kits in the presence of the evasion technique in Section 6 and understand how real phishing kits are being used in the wild by clustering kits with the scrapped webpages in Section 7.

4.2 Phishing Kits

We also collect phishing kits used in the wild. When our crawler visits the phishing websites, it also attempts to retrieve phishing kits in the phishing web servers. The main challenge of collecting phishing kits used in the wild is that we have no specific information about phishing kits' locations and file names on a phishing website. To this end, our approach leverages the fact that phishing attackers might not remove their phishing kits after deploying phishing websites using the kits, leaving their kits on certain URL paths that are publicly accessible and downloadable [32, 62]. Specifically, our crawler visits all of the collected phishing URLs and checks if the directory listing feature (*i.e.*, directory index) is enabled in the web servers of the phishing URLs. If enabled, the crawler starts downloading all of the files, including compressed or archive files (e.g., zip, rar, tar), images, HTML, JavaScript, etc. For directories, it recursively visits the sub-directories and downloads all of the files in the list. If files or directories are not found in the directory listings, the crawler moves to the parent path segment of the phishing URL. For example, suppose that a phishing URL is 'https://phishing.com/root/example/target/index.html'. If the directory listing is not enabled in '/root/example/target/' and no files and directories are found, it recursively visits its parent path '/root/example/'. Finally, we manually confirm if the downloaded files and directories are related to phishing kits. Note that our crawler checks directories based on the reported phishing URLs, not domains, which prevents accidentally collecting benign content belonging to compromised domains. Additionally, we compare archive files' directory trees to the collected URLs' ones, and only use files, whose filenames match, for our study.

Number of Distinct Kits. Our crawler fetches all phishing URLs newly reported to eCX every 15 minutes and collects phishing kits. From the 6,097 distinct phishing domains, we collect 10,082 phishing kits. We verify interactable distinct phishing kits by the following steps. First, by comparing hashes of phishing kits, we filter out duplicated kits. Second, we build a local Apache server and host each kit on the server based on the fact that phishing kits are typically composed of PHP scripts [32, 45, 53, 62]. Finally, as shown in Table 1, we have distinct 4,153 kits from 3,283 domains, observing a 0.83% hit rate of the total crawled domains. Note that our hit rate is slightly lower than 1.15% in the prior work [55]. As noticed in [55], a hit rate may decrease with time because phishing attackers are likely to make more effort to delete the traces of phishing kits in web servers.

Gaining an Overview of Phishing Kits' Features. In order to gain an overview of what features phishing kits generally provide for attackers, and how the features work inside the kits, we first examine the descriptive scripts (*e.g.*, `readme.txt` and `config.php`). This is because these scripts specifically describe how components of kits are used and configured. Next, we search for commonly used names in descriptive scripts within phishing kits, such as 'manual', 'cfg', and 'config' (for configuration), or 'readme' (*e.g.*, `readme.md`). As a result, we obtain 20,206 descriptive scripts from 1,139 kits, of which 16,811 are `readme.md` files. Due to the huge number of files for manual analysis, we mainly focus on the 664 files found in the root directories.

Phishing Features in Kits. Our manual analysis of descriptive scripts reveals the following three main features of phishing kits: (1) exfiltration and user-interaction, (2) evasion, and (3) web appearance components. *First*, the most frequently appearing description is about 'exfiltration' methods that explicitly explain how phishing attackers can exfiltrate the victims' personal information (*e.g.*, credentials) they submitted to the phishing web servers. Additionally, we find user interaction features enable attackers to configure landing pages to obtain more personal information from victims, such as login failure pages, payment pages, and delivery information pages. The exfiltration/user-interaction features will be discussed further in Section 5. *Next*, the descriptive scripts in the kits reveal how to use 'evasion' techniques, such as redirection of URLs, dynamically-generated URLs, and blocklists, as mentioned in Section 2. We focus on the script-level analysis of evasive behaviors – redirection and dynamically-generated URLs –; excluding blocklists that have been extensively studied in a prior study [53], and assess the real-world impact in Section 6. *Lastly*, 'web appearance' components are also mentioned in the descriptive text files. As kits would have different landing page scripts even if targeting the same brands [61], we can leverage the fact to better understand how many phishing kits are used and deployed for phishing websites by comparing the landing pages from kits with those of real-world phishing websites.

[1] `servesrs-xxxxxxxx[redacted][.]cyou`

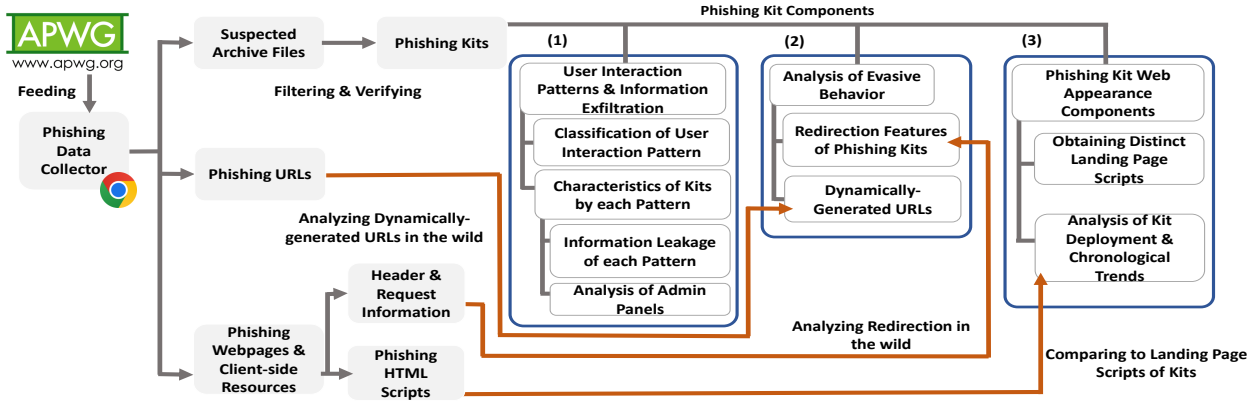

Figure 1: Analysis Overview of Phishing Kit Components and Phishing Sites.

This will be discussed in Section 7. The overview of our analysis is presented in Figure 1.

5 USER INTERACTION PATTERNS & INFORMATION EXFILTRATION

In order to classify user interaction patterns in phishing kits and their utilization of information exfiltrating methods, we analyze the user interaction features of phishing kits with regard to how submitted victims' data are processed among the three entities involved in phishing (victims, kit-deployed servers, and phishing attackers). Then, we identify three user interaction patterns based on how the victims are required to interact with phishing websites and how the victim's personal information is processed inside kits, including the exfiltration of the victims' information. Moreover, in order to gain a better understanding of each classified user interaction pattern, we analyze the differences among each pattern by considering a range of factors, including the information leakage characteristics, targeted phishing brands, and the administrator panels. The administrator panel is a key feature of phishing kits with a more advanced user interaction pattern.

5.1 Classification of User Interaction Patterns

We describe our methodology and classification process for identifying different user interaction patterns of phishing kits based on how the phishing kits process submitted victims' information and interact with potential victims and phishing attackers. For example, the most simple scenario of a phishing website where the website only has a login page, and the victim submits his/her personal information (*e.g.*, credentials). However, a recent phishing kit would offer more advanced features, such as scripts for subsequent webpages (*e.g.*, a login fail page). If potential victims are given the login fail page, in a second trial, they are highly likely to put their real credentials in the phishing login form. We take a step further to examine the interaction patterns at the script levels, showing server-side behaviors such as whether information exfiltration happens in real-time, unlike the previous study [61] that analyzed user interaction patterns between victims and phishing sites by simply checking whether phishing sites have multiple pages.

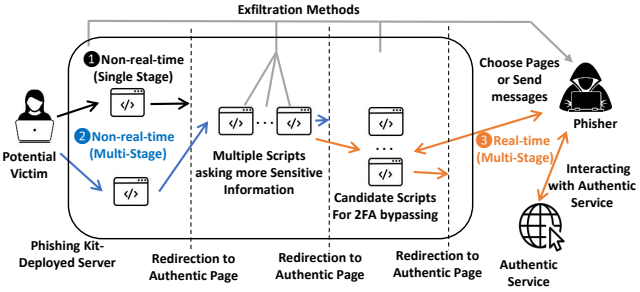

Figure 2: Overview of User Interaction Patterns

Methodology & Classification. To classify the user interactions, we follow the same steps as the victim and track the page navigation by monitoring the destination page of navigating functions like `window.location` or `header`. We also conduct a manual analysis, considering when automated analysis failed. The detailed description is described in Appendix A. As a result shown in Figure 2, we can classify the interactions for 4,153 kits as ❶ *single stage (non-real-time phishing)*, ❷ *multi-stage (non-real-time phishing)*, and ❸ *multi-stage (real-time phishing)*.

Result. We classify three user interaction patterns: one pattern for phishing kits with a single stage, ❶ *non-real-time phishing*, and two patterns for phishing kits with multiple stages, ❷ *non-real-time phishing*, and ❸ *real-time phishing*. *First*, out of 4,153 kits, we find that 1,111 kits (26.75%) have no or single navigation to the original website of the targeted brand. We classify these types of phishing kits as ❶ *single stage (non-real-time phishing)*. We find that this type of phishing kit has only a single landing page script, and any additional communication between the potential victim and the web server is made after the first submission. On the other hand, we check that the remaining 3,042 kits have multiple scripts that the potential victim is navigated to, called *multi-stage phishing* intending them to submit more personal information in detail, as mentioned in the previous study [61]. From the remaining 3,042 kits, we classify 2,881 kits as ❷ *multi-stage (non-real-time phishing)*, where user interaction patterns of the kits navigate accesses of the potential victims to additional scripts, while the kit-deployed servers exfiltrate data submitted to phishers.

#Stage	User Interaction Pattern	#Kits	Avg. # of Info. Types	Avg. # of Interaction	Exfiltration Methods Found(%)
Single stage	❶ Non-real-time	1,111	2.61	1.06	Mail: 881 (79.29%), Telegram [20]: 9 (0.8%), Local File: 324 (29.16%)
Multi-stage	❷ Non-real-time	2,881	3.79	1.78	Mail: 2,281 (85.49%) , Telegram: 600 (20.83%), Admin Panel: 580 (20.13%)
	❸ Real-time	161	4.25	3.98	Admin Panel: 161 (100%), Mail: 141 (87.58%), Telegram: 35(21.74%)
Total		4,153	3.49	1.67	Mail: 3,303 (79.53%), Admin Panel: 741 (17.84%), Telegram: 644 (15.51%)

Table 2: Information Leakage Characteristics of Phishing Kits

During the manual analysis of descriptive texts in Section 4.2, we discover that some phishing kits offer real-time interaction by combining administrator panels and specific plugins, such as Jabber [59], the XMPP protocol for real-time interaction. Additionally, we find a hidden user interaction pattern that facilitates 2FA bypassing, navigating potential victims to additional scripts within the phishing kits, where the phisher has curated scripts to either navigate the potential victim or act as a man-in-the-middle attacker with the intention of bypassing two-factor authentication in real-time [41]. We classify these types of kits as ❸ *multi-stage (real-time phishing)*. In the pattern, the phishers not just receive data from the server but also send data to the server in real-time, which is not recognized by the potential victim. In order to enable these functionalities, it is crucial for the kits to have a library or plugin that provides real-time interaction. By leveraging this fact, we identify 161 kits including scripts of real-time interaction plugins and libraries.

5.2 Characteristics of Kits Based on User Interaction Patterns

To classify phishing kits based on user interaction patterns, we first analyze the characteristics of each pattern based on information leakage flow inside the kits. For this purpose, we measure the types of target information, validation, and exfiltration methods used in each class. We also analyze how user interaction patterns of phishing kits are related to the authentic target brands. Additionally, we deeply analyze the administrator panel, an advanced exfiltration method, which plays a significant role in the real-time interaction of ❸ *real-time phishing*. We describe what components constitute administrator panels and analyze multi-brand panels and real-time plugins of existing frameworks.

5.2.1 Information Leakage of User Interaction Patterns. Due to different steps of user interaction patterns, each type has a different range of information leakage. In order to assess the information leakage of each user interaction pattern, firstly, we measure the personal information types that a phishing kit targets. Secondly, we measure how often the kit tries to navigate potential victims to gather more sensitive information. Finally, we measure the diversity of the exfiltration methods of each pattern, such as mail, Telegram bots, and administrator panel.

To measure the three aforementioned factors, we leverage the fact that phishing kits tend to have specific scripts for dealing with exfiltrating information. As shown in Listing A2 in Appendix, we can collect the types of information submitted by values written on the scripts. Specifically, by using regular expressions and classifying information with keywords, we first collect 20 types of information submitted to kits based on our empirical analysis, as shown in Table A4 in Appendix. In addition, we find that phishing

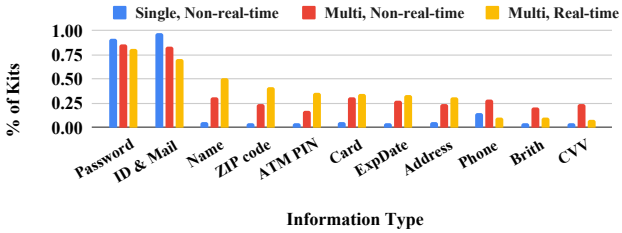

Figure 3: Exfiltrated Information Type

kits tend to exfiltrate submitted information at every step when navigation happens. Thus, we secondly measure the number of interacting steps dealing with the submitted information to find the number of attempts to validate submitted information by the kits during the user interaction. To figure out which exfiltration method phishing kits are using, we finally examine each kit to see if it uses the `mail()` function and includes mail addresses in PHP scripts. Additionally, we search for PHP scripts in the kits that include the Telegram bot (*e.g.*, https://bot.telegram.~) and confirm whether there is a token and a bot key. To find admin panels further, we check whether filenames include 'admin' or 'panel' using regular expressions and manually analyze if the kits we found show the submitted information on the browser.

Results. We analyze each kit type based on the information leakage characteristics we measured, specifically the number of information types, the number of interaction trials for validating information, and the diversity of exfiltration methods, as shown in Table 2. Figure 3 shows the information types each kit type can handle (*i.e.*, submitted). The most frequently submitted information types are *IDs & mail addresses* and *passwords* for all user interaction patterns. However, specific information types, especially for more sensitive information, have significant differences in user interaction patterns. For example, 36.11% of ❸ Multi-Stage (real-time phishing) intend the potential victim to submit *ATM PINs*, while only 17.58% of ❷ Multi-Stage (non-real-time phishing) ask for ATM PINs. This is because most of the ❸ kits target financial brands (in-depth analysis of it will be given in Section 5.2.2). From the perspective of exfiltration methods, we find that most ❸ phishing kits use multiple methods simultaneously. Phishing kits with admin panels, which are the most commonly used method for ❸, have various ways to compose the visual webpage. Of the 741 (17.84%) panels we analyzed, 497 admin panels read text files in which the submitted information is written, while the other 244 panels read the submitted information from databases such as SQLite3 inside existing frameworks.

5.2.2 Targeted Brands. We measure the targeted brands of the phishing kits to determine the relationship between user interaction patterns and the brands they are targeting. To this aim, we examine the brands of the kit-found domains from the APWG reports. If multiple kits are found in one domain, we manually check the

Framework	Technology	#Kits	Framework	Technology	#Kits
Laravel [24]	Redis [7]	28	uAdmin	Jabber	25
	WebSocket [33]	11	unknown	WebSocket	5
	Laravel-pusher [17]	4	Drupal [2]	Redis	5
	Laravel-echo [10]	2	Twilio [15]	Twilio Chat*	4
CodeIgniter [13]	Redis	33	Bootstrap [16]	WebSocket	2
	LiveChat [5]	4	Dolibarr [4]	Redis	1
	Jabber [59]	1		Mercure [23]	1
WordPress [12]	Redis	18	Joomla [11]	WebSocket	2
	Sendinblue [19]	7	Moodle [6]	WebSocket	2
	WebSocket	4	gRPC [22]	Streaming bot	1

*: Twilio Programmable Chat

Table 3: Existing Frameworks and Real-time Technologies in Admin Panels

landing page scripts, images, and favicon files that indicate which brands are targeted. As shown in Table A1 in Appendix, Microsoft is found to be the top targeted brand in all of the three patterns. In detail, Adobe is the second most targeted company in the ❶, but the percentage decreases significantly in the ❷, and there are only two kits in the ❸. Moreover, fake email brands are not observed only in the ❸; rather the phishing kits in ❸ tend to target various financial brands (Wells Fargo, PayPal, Chase, etc.) as well as online business services.

5.2.3 Analysis of Administrator Panels. To better understand how admin panels work and the relationship between the panels and user interaction patterns, we focus on the key functions of admin panels. Admin panels in phishing kits provide two key functions: (1) options for choosing targeting brands, which helps phishing attackers easily configure kits (*e.g.*, some admin panels offer additional landing page scripts of other brands and switch hosting providers by simply checking the options), and (2) an extra exfiltration method, especially for real-time interaction. Although there is a panel that only tells phishers the current status of exfiltrated information with a single script, advanced phishing kits leverage existing frameworks to provide real-time interaction through plugins.

Multi-Targeting Brand Panels. During manual analysis of descriptive texts in Section 4.2, we find that the specific descriptive texts explicitly mention their phishing kit identities, `uAdmin`, `Spox`, `XBALTI`, and `NewOreoo`, have admin panels that offer multiple target brands. To figure out the exact number of phishing kits that cannot be found from descriptive texts in our dataset, we further search phishing kits for their names or identify fingerprints of each kit by their specific file names and strings inside their scripts. For example, `uAdmin` has its own directory named 'a1b2c3', and the string 'uadmin' appears inside the scripts of the same kits. We identify 214 kits (5.15%) of multi-targeting brand panels. Specific trends of each multi-targeting brand panel are shown in Appendix B. Although such phishing kits with multi-brand panels are the minority of the total kits, as the number of targeting brands increases, anti-phishing agents will face greater difficulty in determining the true identity of the attacker responsible for the phishing campaign and the extent of the attack.

Usage of Real-Time Interaction Technologies. We find that the `uAdmin` phishing kit offers a single remote admin panel hosted on a remote web server to control multiple phishing websites, even if their target brands are various. This functionality works with real-time interaction plugins, an essential component to help bypass 2FA,

```
1     ...
2     <h4 class="waiter__h4">\
3         Please wait...\
4     </h4>\
5     <p class="waiter__p">\
6         We are checking your information\
7     </p>\
8     ...
9  REST_FN__ = {
10    ask_login: {
11        send: function(data) {
12            CORE__.show_wating();
13            respond.prev_op = "ask_login";
14            ...
15            CORE__.send_home(function() {});
16        },
17    },
18    ask_otp: {
19        ...
20    },
21 };
```

Listing 1: Example of Real-time Interaction Features in `uAdmin`. If information submission is completed on a single page, the kit shows a fake subsequent waiting page (lines 2-7). Then, the kit sends information submitted to the phisher with prearranged opcodes such as "ask_login" or "ask_otp" and waits for the subsequent message from the phisher (lines 10-16).

as shown in Listing 1. These technologies are integrated into phishing kits directly through web frameworks like WordPress [12] and Laravel [24], without requiring the phishing kit creators to develop the technologies from scratch. To determine which real-time interaction technologies are being used, we examine the configurations and backend infrastructure of each phishing kit.

The results in Table 3 indicate how advanced these phishing kits have become. Of the 161 ❸ multi-stage (real-time) phishing kits we examine, the most popular real-time interaction technology is Redis [7] – an in-memory data store and caching layer for web applications – that provides a publish/subscribe mechanism to interact with users in real-time. This in-memory framework decreases delays between potential victims and kit-deployed servers, making the phishing website look more authentic. Phishing kits with Redis typically include scripts for subscriptions when users access phishing sites. Once a user subscribes, the kit using Redis will publish user data to the remote server where the attacker is located, allowing real-time interaction between the attacker and the victim. Moreover, the framework offers *live chats* to facilitate real-time interaction.

The next most commonly used real-time interaction technologies are Jabber and WebSocket, each of which is found in 26 kits. We assume that `uAdmin` is a type of framework because phishing kit creators have created it without using the other existing frameworks. All kits using `uAdmin` interact with phishers by Jabber, an instant message protocol called XMPP. Interestingly, real-time interaction scripts are not found in 5 `uAdmin` kits, while configuration scripts of them still have options to configure how to use Jabber. Instead, they are using the Telegram bot as the information exfiltrating method, showing the flexibility of the advanced phishing kits. Unfortunately, we cannot identify the framework of 5 kits using WebSocket; all script names inside the kits are obfuscated.

Takeaway. We identify submitted information types and attempt to validate information increase in the order of ❶ single Stage (non-real-time), ❷ multi-stage (non-real-time), and ❸ multi-stage

(real-time) phishing kits. Additionally, by analyzing administrator panels, we find 214 kits (5.15%) provide multi-brands, and 161 kits (3.88%) can be classified as ❸ real-time phishing, because of their real-time interaction technologies. We also find that a real-time interaction technology, Redis, is most frequently used due to its low latency and the ability to provide live chat functionality.

6 ANALYSIS OF EVASIVE BEHAVIOR

The evasive behaviors provided by phishing kits are one of the essential functionalities for phishing attackers to have anti-phishing bots and agents challenged to access the phishing websites, which helps extend the lifespan of phishing websites.

We focus on the prevalence of the two evasive behaviors – (1) redirection of URLs and (2) dynamically-generated URLs –; *first*, to deeply understand how the (1) *redirection* of phishing domains and URLs occurs, we analyze redirection features inside phishing kits and measure how many real-world phishing websites have the redirection features in the phishing ecosystem. *Second*, we focus on how phishing kits (2) *dynamically-generated* URLs. Moreover, we measure what dynamically-generated URL patterns are created by kits and find out which patterns of phishing sites' URLs are in the wild.

```
1  [config.php]
2  <?php
3  ...
4  $autoEmail="true";//"true":Auto email grabing "false":
       without auto email grabing
5  $pagelink="http://domain.com/OfficeV4";
6  $FailRedirect="https://www.wikipedia.org/wiki/
       Microsoft_Office";
7  $redirecttype="2";// 1:header - 2:script
8  ...
9  ?>
10
11 [index.php]
12 <?php
13 ...
14 echo "<script type=\"text/javascript\">window.location.
       href = \"$pagelink\"</script>\n";
15 ...
16 echo "<script type=\"text/javascript\">window.location.
       href = \"$FailRedirect\"</script>\n";
17 ...
18 echo "<script type=\"text/javascript\">window.location.
       href = \"$pagelink/$email\"</script>\n";
19 ...
20 ?>
```

Listing 2: Example of Redirection Features in the Kits

6.1 Redirection of Phishing Domains & URLs

As described in Section 4, in our collected dataset, we observe many cases where initial phishing URLs are redirected to other URLs, which helps evade the current blacklist-based anti-phishing defense mechanisms [37, 49, 52]. We aim to analyze how redirection features are implemented inside kits and how the features are used in real-world phishing websites.

Redirection Functions & Destinations of Kits. To get an insight into how redirection works, we randomly select 20 phishing kits with redirection features from our manual analysis of descriptive texts in Section 4.2. We analyze the kits at the script level and identify several significant attributes. *First*, each kit has a configuration script that contains two links: one for redirecting victims to the original phishing website and the other for redirecting them to the authentic website of the targeted phishing brand if access is filtered. For example, in Listing 2 (lines 14 and 16), the kits redirect accesses to each link by using the JavaScript function `window.location.href`, which returns the page of the URL. Moreover, as seen in Listing 2 (line 4), the kit provides an email-grabbing option. This option identifies emails from the URI of incoming access and redirects access to URLs containing email (line 18 in Listing 2). It implies that phishing attackers, using email-grabbing, attack only if the request URI contains the victim's email address that the phishing attacker targets.

Category	Total #Kits (%)	Category		Total #Kits (%)
HTTP Status Code	522 (19.98%)	Auth. Website	Microsoft	245 (9.38%)
Variable or Empty Value	1,443 (55.22%)		Google	117 (4.48%)
			Financial Brands	61 (2.33%)
Filepath of Kit	1,161 (44.43%)		Others	192 (7.34%)
			Total	615 (23.54%)

*: Wells Fargo, **: M&T Bank

Table 4: Input Values of Redirection in 2,613 Kits

Next, we analyze the index PHP and HTML scripts of phishing kits to extract redirection features. We collect 13,875 index PHP and HTML scripts from 4,153 kits and examine the usage of eight popular redirection-related functions in PHP, HTML, and JavaScript. As a result, shown in Table A3 in Appendix, we find seven functions (1 of PHP, 1 of HTML meta tag, and 5 of JavaScript) from 5,926 scripts in 2,613 kits (62.92% of the total 4,153 kits). The `header` function in PHP has been identified as the most frequently used function in a single script, with a maximum of 47 occurrences due to various redirection conditions. Additionally, 1,074 HTML scripts from 707 kits (17.02%) utilize redirection functions, indicating that anyone accessing these HTML scripts in the wild can easily identify the presence of redirection functions, unlike in PHP scripts.

We collect input values of discovered redirection functions to determine the destinations of phishing kit redirections. These values are categorized into four groups: HTTP status code responses for connection errors, variables or empty inputs for kit customization, redirection within kits, and authentic websites. The analysis in Table 4 reveals that variable or empty inputs are the most common values, indicating that kit users are responsible for setting the redirection links. Notably, phishing kits respond with specific HTTP status codes (404 or 403 errors) when accessing from a blocklist. Additionally, we find a limited number of specific authentic websites to which initial access is redirected. Microsoft services (Office, OneDrive, and Outlook) rank highest, followed by Google services (Gmail). This observation suggests that redirecting initial access to specific authentic websites can be used as a feature to help identify phishing sites.

URL Redirection in the Wild. To see how the redirection features are used in the phishing ecosystem, we compare the list of URLs initially reported to the APWG eCX with the URLs of the landing pages that our crawler has finally landed. As shown in Table 5, of the total 14,373,422 accessed URLs, we find 273,440 (1.90%) redirections to other domains and 1,414,491 (9.84%) redirections to other URLs in the same domain. Specifically, 17.42% of the 273,440 domain redirections are to the other phishing domains reported in APWG eCX, not to other authentic domains. The rest of the domain redirections (82.57%) are confirmed to domains of authentic, legitimate services (*e.g.*, microsoft.com), which means the phishing sites may recognize

Redirection Type	#Domains Redirection To			#URL in Same Domains (%)
	Authentic Do. (%)	Phishing Do. (%)	Total (%)	
Only Desktop	3,036 (1.11%)	1,060 (0.39%)	4,096 (1.49%)	22,544 (1.59%)
Only Mobile	2,796 (1.02%)	1,625 (0.59%)	4,421 (1.61%)	51,619 (3.65%)
Both	219,973 (80.44%)	44,950 (16.44%)	264,923 (96.88%)	1,340,328 (94.75%)
Total	225,805 (82.57%)	47,635 (17.42%)	273,440 (100%)	1,414,491 (100%)

Table 5: Redirection of Phishing Websites

URL type	#Domains(%)	#URLs (%)	Examples
Random Names of Directories	33 (36.26%)	5,776,004 (50.44%)	a.com/PRbdEEy30Btfgt5yv/
Incremental Number	28 (30.77%)	2,802,770 (24.47%)	a.com/1644307804/login
Random HTTP GET Values	21 (23.08%)	1,802,672 (15.74%)	a.com/?login=TZaQ&user=UpM3
Recursive Generated Directories	5 (5.49%)	366,872 (3.11%)	a.com/siteassets/bank/siteassets/bank/...
Random Hex Strings	2 (2.20%)	665,401 (5.81%)	a.com/#%08%C3%A0%C2%AA%CB
Random Names of Scripts	2 (2.20%)	36,618 (0.32%)	a.com/6LRZM9.php
Total	91	11,450,337 (100%)	-

Table 6: Various URL Generation Types for Phishing Domains with More Than 10,000 URLs

our crawler not as the potential victim and redirect our crawler to authentic, legitimate services (*e.g.*, microsoft.com, adobe.com, and DHL.com). Additionally, 3.01% of domain redirections target only mobile and desktop environments. We suspect that the domain redirections have occurred because the reported phishing domain links have been delivered as desktop or mobile-dependent services, and the phishing attackers considered the environment.

6.2 Dynamically-Generated URLs in the Wild

In our collected dataset, we have observed that several phishing domains have an excessive number of different URLs. We have also observed that such different URLs are dynamically generated for each victim. For example, multiple URLs in the same domain differ only in one directory or script name as randomized strings with the same length. Such dynamically-generation features for URLs can help bypass the blacklist-based anti-phishing defense mechanisms such as Google Safe Browsing (GSB) [51]. To better understand these dynamically-generated URLs features as an evasion technique, we first analyze the features in our collected phishing kits; specifically, we seek to know how the features are implemented in kits and how they are used in the wild.

Six Patterns of Dynamically-Generated URLs. We characterize dynamically-generated URLs and observe that the kit explicitly implements dynamically-generated URLs components in the index pages (*e.g.*, `index.php` or `index.html`). In these index files, when access requests (*e.g.*, from potential victims or anti-phishing agents) are sent to phishing websites, the PHP scripts call `mkdir()` or `rand()` functions, each of which makes directories and randomizes numbers. For example, as shown in Listing A3 in Appendix, a random string is generated in line 3, which uses `rand()` function in line 28, the original main script or directory is copied in line 5, and, at last, the `header` function, which redirects access to the input URL, is used in line 11 to redirect to a newly-generated URL. By leveraging this fact, we check if either one of the functions is on scripts named 'index', such as `mkdir()` and `rand()`; and, if this is the case, we further check if their outputs are used as inputs of redirection functions. As a result, we find the existence of dynamically-generated URLs from 2,091 kits.

To characterize dynamically-generated URLs, we analyze URL-generating functions from the index scripts of 2,091 kits. We focus on the functions used to create directories, scripts, or randomized strings for URLs such as `mkdir()`, `copy()`, `rand()`, and `hex2bin()`, and their input values. As a result, we identify six common patterns of phishing URLs from kits by analyzing which parts of the URLs are dynamically-generated, as shown in Table 6. Each of the six patterns we identify in our dataset employs various patterns to create unique URLs for phishing websites: random names of directories, incremental number, random HTTP GET Values, recursive generated directories, random hex strings, and random names of scripts. Three of these patterns involve randomizing the names of directories, scripts, and HTTP GET values. One pattern takes a step further by encoding randomized names of directories or scripts as hex strings. The other two patterns do not use a random function but rather employ incremental numbering as directory names or recursively copy their directories, resulting in extremely long URLs.

To figure out how these six patterns are shown in the wild, we select 91 phishing domains with more than 10,000 URLs (10.5M URLs) to better observe the use of dynamically generated URLs. As shown in Table 6, the three most common URL patterns – random names of directories, incremental number, and random HTTP GET values – are used by more than 90% of domains and URLs. This implies that anti-phishing agents can assess the phishing suspicion level of a domain by checking if these patterns are frequently used in URLs. For example, as shown in Table A5 in Appendix, we have analyzed the URL patterns of the authentic login pages of the top 30 brands identified from phishing URLs. Among them, 14 brands have HTTP GET values patterns in their URLs, 10 have directory patterns, and 6 have script names with extensions at the end. The authentic login pages' script names, such as 'login.jsp' and 'client.php,' are not arbitrary but rather have names suitable for their function. This holds true for brands with directory patterns as well.

Bogus Parameters. Notably, we decide to conduct a detailed analysis of the third most common pattern, which shows HTTP GET parameters of phishing sites. Unlike other patterns, kits can use their HTTP GET values, which gives us more insight into the correlation between phishing URLs and kits. To figure out how they use HTTP GET values by investigating kits, we analyze 1,350 kits with random values in their URLs, and we search for their parameter names in the scripts of kits extracted from those domains. Our findings indicate that 1,187 kits generate URLs with bogus HTTP GET form data, while only 163 kits collect HTTP GET values from URLs. The detailed usage is given in Appendix C

Takeaway. We identify six common patterns of dynamically-generated URLs from the kit analysis. We observe that phishing sites actively use them and find that the three most common patterns – random names of directories, incremental number, and random HTTP GET values – are used more than 90% of phishing domains with more than 10,000 URLs. Through a detailed kit analysis of the third most common pattern involving HTTP GET parameters, we find that most phishing kits generate URLs with bogus HTTP GET form data rather than collecting HTTP GET values from URLs.

7 PHISHING KIT WEB APPEARANCE COMPONENTS IN THE REAL WORLD

In this section, we extract landing page scripts from phishing kits to figure out the distributions of their deployments in the real world. By finding the similarities between the HTML codes of phishing sites and those of the landing pages created by phishing kits, we

can establish a relationship between the deployments of the kits and the phishing sites in the wild. Note that we utilize our two collected datasets (*i.e.*, phishing kits and phishing websites' client-side resources) for the clustering.

7.1 Distinct Landing Page Scripts & Similarity Matching

To obtain the index HTML code from phishing kits, we install/deploy phishing kits in an experimental environment (Apache web server on Ubuntu 18.04 with the PHP 7.2.24.) simulating real-world phishing websites, as shown in Figure 4. Specifically, we extract the HTML code of the landing pages from our own deployed phishing websites using Selenium Chrome WebDriver. Then, we compare the landing pages from the deployed phishing kits with the landing pages of the client-side web resources of phishing websites used for real phishing attacks.

Although the main PHP script, `index.php`, works as a landing page script in general cases, this is not the only case in the phishing kit; instead, in the phishing kits, it mainly works as a script for operating evasion techniques such as dynamically generating URLs and the other script, such as `login.php`, works as a landing page script showing fraudulent login page to the potential victim. Thus, we select not only the main scripts of the phishing kits but also potential landing page scripts containing keywords (*e.g.*, `index.php`, `login.php`, `index.html`) in their file names from the upper paths of phishing kits. We determine the number of candidate scripts to be 10 based on our empirical analysis. The analysis shows that most of the landing page scripts of phishing kits are located in the root path, and the number of scripts located in the root path is mostly less than 10. We then visit each candidate landing page using a Selenium Chrome WebDriver [18]. We use the `fake-useragent` [21] library in Python to avoid the evasion technique of phishing kits.

Similarity Matching. After obtaining landing pages from two datasets (phishing kits and phishing websites' client-side resources), we then score the structural similarities between the original HTML script of each phishing website and the candidate landing page scripts obtained from the phishing kits. The reason for utilizing structural similarity is that the JavaScript in the landing pages could be different even though both pages are visually identical. For example, we empirically observe that some phishing kits have visually identical landing page scripts but have different JavaScript functions for processing victims' personal information. Moreover, some phishing kits obfuscate specific values of their HTML scripts for each phishing URL with the same domain, such as `<a class = 'TC9skcl4Kz'>`, or encode all contents in them. This makes it difficult for the other similarity methods to verify if the index HTML scripts are from the same phishing kit, although there is no visual difference on the browsers.

We use the simple sequence comparison algorithm [67] of HTML tags for structural similarity considering the computational time, especially in the case where there are hundreds of tags (this is the case of our collected HTML scripts). We determine a 75% similarity score as a threshold for candidate scripts because we confirm visually identical scripts have over 75% structural similarity based on our empirical analysis and the previous studies that use the number as the threshold [46, 57, 58]. After pruning error pages

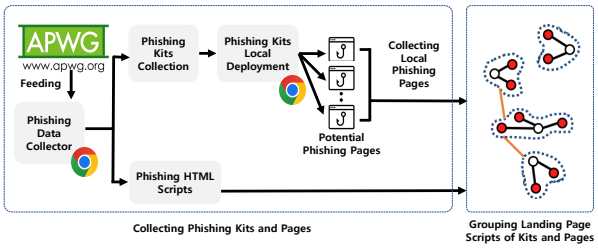

Figure 4: Pages Collection and Grouping of Phishing Kits

User Interaction Type	#Scripts	#Kits	#Domains(%)	#Domains/#Scripts
❶ Single stage (non-real-time)	155	576	26,625 (28.08%)	171.77
❷ Multi-stage (non-real-time))	502	1,333	64,100 (67.59%)	127.69
❸ Multi-stage (real-time)	20	112	4,109 (4.33%)	205.45
Total	677	2,021	94,834	140.08

Table 7: Kit Deployment by User Interaction Types

in the phishing webpage dataset, such as '404 not found' pages, finally from 2,021 kits, 677 distinct index page scripts are identified. Each index page script is originally from, on average, 2.99 kits. The largest number of sharing an index page script is 172 phishing kits that mimic Adobe page where there is the login form of Adobe website with Microsoft IDs and passwords (*i.e.*, Microsoft Single Sign On); followed by the second largest number, 84 phishing kits that mimic the Microsoft login pages. Each of the rest 431 index page scripts is from only a single kit.

7.2 Measuring Phishing Kits in the Wild

HTML Scripts from Real-World Phishing Sites. To compare the HTML scripts extracted from the phishing kits with those from real-world phishing websites, we first group the HTML scripts extracted from the phishing websites. We select 91 domains with more than 10,000 URLs, a total of 11,450,337 (11.4M) URLs, and check how many different URLs can be grouped together for each domain based on the structural similarity. As a result of our analysis, we recognize that most of the URLs with the same domain have the same index HTML scripts. For example, we observe the domain `actvess-xxxxxx[redacted][.]xyz` has 11,417 different index HTML scripts, but all of them are structurally the same. As a result, we extract 91 HTML scripts from them using the same process for each domain. Despite the numerous URLs, the maximum number of different HTML scripts from one domain is 9, and the single script tends to be deployed in more than 90% of URLs of one domain, showing the prevalent deployment of the dynamic URL evasion technique. By conducting the same analysis on the rest of the domains having less than 10,000 URLs, we also extract one representative HTML script among URLs in each phishing domain; and, finally, extract 497,856 index HTML scripts from all of the 497,856 domains.

To discover the distribution of phishing kits in the wild, we finally compare 677 HTML scripts to 497,856 index HTML scripts of phishing websites. As a result, 94,834 index HTML scripts of phishing domains, about 19.05% of the total 497,856 phishing domains, are structurally identical to some of the 677 HTML scripts extracted from phishing kits. This implies that a single landing page script is used for 140 domains on average, highlighting the fact that a non-negligible number of domains show the same phishing pages.

Trend	#Scripts (#Kit)	#Do.* (#Do./#Kits)	#Scripts by Interaction (#Do.)
Regular	75 (855)	61,800 (72.28)	❶: 19 (20,409), ❷: 54 (40,996), ❸: 2 (385)
Bursty	15 (117)	8,236 (70.39)	❶: 3 (1,468), ❷: 9 (3,653), ❸: 3 (3,125)
Total	90 (972)	70,036 (72.05)	❶: 22 (21,877), ❷: 63 (44,649), ❸: 4 (2,127)

*: Domains

Table 8: Bursty and Regular Phishing Kits

Targeted Brands of Kit-deployed Domains. We conduct an analysis of phishing kit-deployed domains to identify trends of targeted brands in the phishing kits. As also previously reported in [61], we find multiple landing page scripts exist for a single brand, which means there are multiple kits with different landing page scripts. Our investigation reveals that, on average, each kit with the same single landing page script is deployed in 46.92 domains, showing the average number of potential deployments a single kit has.

In the top 6 targeted brands of kit-deployed domains shown in Table A6 in Appendix, we observe that 146 scripts are used for phishing Microsoft, and the Microsoft - Adobe Generic (MAG 1) script is the most frequently deployed with 6,939 domains. The Facebook landing page script (Facebook 1) is the second most frequently deployed, with 3,881 domains. The Microsoft landing page script (Microsoft 1) is deployed in 3,393 domains, ranking third in frequency. Among the top 6 targeted brands, we also find financial institutions such as Chase and Wells Fargo, as well as e-commerce services such as Amazon.

User Interaction Types of Phishing Kits in the Wild. As shown in Table 7, we check how many phishing kits of which user interaction types are deployed on domains. Even if there are fewer domains deployed by ❸ kits, their impact on the phishing ecosystem is non-negligible, considering the number of domains is more than the average number of domains deployed per script. In addition, ❷ kits are most widely deployed on domains at present. Compared to previous research [61], which has discovered that about 45% of phishing sites use multi-stage phishing, we discover that about 70% of kit-deployed domains use multi-stage phishing kits.

7.3 Chronological Trends of Phishing Kits

We measure how each kit-deployed domain changes on a monthly basis during our observation period (Aug. 22, 2021 – Jan. 31, 2023). We especially focus on investigating what kind of kits radically reach their peaks and fall in a very short period of time, which is called a *bursty* kit. We then analyze which brands and phishing kits have such trends and discuss their implications in practice.

Bursty Phishing Kits. To classify phishing kits into bursty kits and regular kits, we first select 90 landing page scripts deployed in more than 10 phishing domains per month during the period. We then check each index script to see if the number of deployed domains exceeds 20% of all deployed domains in some months. As a result, we find 15 specific landing page scripts (117 kits) showing the bursty pattern, while 75 (855 kits) are consistent. Figure 5 shows the top three landing scripts of each user interaction type. As shown in the figure, each pattern has a huge difference.

Table 8 shows that the number of bursty kits is less than that of regular kits. However, considering the conspicuous number of deployed domains in a short period, their practical impact is no less than that of a regular kit in the period. For instance, one of the identified bursty kits showed a peak of 416 deployed domains in one month. In contrast, the regular kits only had an average of 10 domains per month, demonstrating the potential impact of bursty kits in the phishing landscape. Among the landing page scripts, Microsoft scripts dominate the regular patterns. However, none of the scripts imitate Microsoft for busty patterns. Instead, six scripts mimic Wells Fargo, and one script from M&T Bank stands out with 1,995 deployed domains. Furthermore, 10 of 15 landing scripts primarily target banking brands, including Wells Fargo, M&T Bank, ITAÚ, Commonwealth Bank of Australia, and Alaska Federal Credit Union. On the other hand, three scripts focus on social networking services (2 Facebook and 1 Instagram). This insight sheds light on the more dangerous brands for short-term phishing attacks.

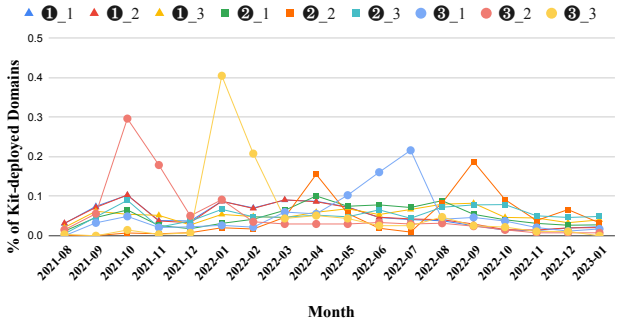

Figure 5: Monthly Ratio of Kit-Deployed Domains by Each Top Three Landing Page Scripts. The top three landing scripts of ❸ kits are radically increased in specific periods showing bursty patterns, while the other ❶ and ❷ kits show relatively even distribution.

Takeaway. We find one kit is deployed in an average of 46.92 domains, showing the average number of potential deployments a single kit has. We also disclose that multi-stage kits account for more than 70% of kit-deployed phishing domains, and about 5% of the domains deploy real-time phishing kits. Finally, we identify 117 kits that show a bursty deployment trend, mostly targeting financial brands, especially banks. Consequently, we confirm that phishing kits are evolving into conducting more sophisticated attacks with real-time user interaction in a hit-and-run fashion, especially for financial gain.

8 DISCUSSION

8.1 Limitation

In this study, we aim to measure the prevalence and impact of phishing kits in the wild. However, our study has the following limitations in terms of data collection and analysis. Firstly, we can collect kits only when the kits are not removed and left on the reported URL paths. If a phishing kit has a function to delete itself after unpacking or is manually deleted by the attacker, we cannot collect it. Therefore, diversifying phishing kit gathering methodologies that can overcome such a challenge could help provide a better understanding of phishing kits' behaviors. Secondly, we can identify real-time phishing kits potentially used as 2FA bypassing by investigating their interaction technologies as well as JavaScript with WebSocket functions. However, we could not analyze JavaScript when it was obfuscated, which happens in the non-negligible number of the kits. Thus, finding more effective evidence or features

for identifying real-time phishing kits and other kit types in the presence of such obfuscation mechanisms remains a challenging and open problem in the phishing kit analysis literature.

8.2 Contributing to Phishing Detection

Web Interaction. We discuss how our research enhances web interaction-based phishing detection by applying our findings to the state-of-the-art reference-based detection method [45], which is based on the recording reported pairings (*i.e.*, phishing domain - brands identified using OCR-based techniques). When the detection fails to determine the website's brand, it submits fake information to the website and checks if the submitted information is identified as false by the website. In this context, the detection model arbitrarily sets the number of interactions during information submission. We observe that the ❸ multi-stage real-time kits with the highest instances of interactions typically have an average of more than four interactions. Our observation helps the detection model determine an appropriate interaction threshold. Additionally, a time threshold for fake information submission can be precisely set by considering the potential delay in real-time interactions where phishers use administrator panels to verify user information, as mentioned in Listing 1.

URL Patterns & Redirection. Our analysis in Table 5 reveals that 20% of domain redirections caused by phishing websites navigate potential victims to other phishing domains. The finding highlights that anti-phishing bots need to verify not only the access URL but also the final destination URL to counter the entire phishing attack process effectively. Moreover, in cases where URL redirection occurs, detection models can heighten the suspicion level of the phishing URL by utilizing the phishing URL generation pattern in Section 6.2.

Kit-Deployment Trends. As shown in Table 8, we find that the majority of ❸ real-time phishing kits exhibit a deployment trend characterized by a hit-and-run fashion. This finding can contribute to exploring and preventing entire phishing campaigns rather than solely focusing on the detection of individual phishing sites. Assume that when an anti-phishing agent is detecting a phishing site, it leverages features specific to real-time phishing kits, such as delays between interactions or the use of particular web frameworks, to identify if a real-time phishing kit has generated the site. The anti-phishing agent can proactively notify the targeted brand about the bursty deployment trend of real-time phishing kits, issuing a preemptive warning to the brand's consumers and accelerating the early detection of phishing campaigns.

9 RELATED WORK

Recent studies analyzed components of phishing kits. Especially for evasion techniques, since *.htaccess* was analyzed in [53], how server-side [51, 52] and client-side evasion techniques [70] influence the anti-phishing ecosystem have been investigated. In addition, while numerous recent studies were proposed for phishing detection [25, 29, 34, 44, 45, 48, 65, 69] and data exfiltration [27, 31, 55], only a recent study [61] has investigated multi-stage phishing, aiming for a better understanding of user interactions in phishing sites. However, this study focused only on user interactions of phishing sites, while our research offers a more comprehensive view, including server-side behaviors such as real-time interaction, by scrutinizing phishing kits at the script level.

Script-Level Analysis on Phishing Kits. Cova et al. [32] first identified the existence of ready-to-deploy PHP written packages, called phishing kits nowadays, and showed snippets of obfuscation and backdoor exfiltrating data from the phishing website. Merlo et al. [50] analyzed PHP, JavaScript, and HTML codes from 20,871 kits using static similarity analysis. They reported that 90% of the kits share 90% or more of their source codes with another kit. However, the study did not delve into the specific functionalities associated with the shared source codes. Tejaswi et al. [62] collected 4,238 phishing kits and identified backdoors, information exposure, and security vulnerabilities within them. They discovered obfuscated emails through dynamic analysis tools for PHP and hard-coded Telegram bots, exposing significant stolen data obtained by phishing kits. Compared to their studies, our research focused on statically analyzing how scripts are implemented for evasion techniques and the extent to which they are shared in the wild.

Phishing Kits in the Wild. Han et al. [36] used a sandbox approach to lure phishers to deploy phishing kits on their honeypot servers and analyzed their lifecycles such as duration and locations. Bijamins et al. [28] fingerprinted Dutch phishing kits, clustered phishing kit families with features found in directories and codes, and searched fingerprints of kit families in the wild. Compared to their research, we use structural similarity instead because of the large number of phishing kits to use unique strings as fingerprints. We also analyze kit deployment from the worldwide phishing landscape. Kondraki et al. [41] analyzed three MITM phishing toolkits that proxy user traffic to bypass 2FA in real-world scenarios using network-level properties and proposed a resilient detection scheme. In our study, we analyzed real-time phishing kits, which are also capable of 2FA bypassing in a broad sense by real-time interaction among potential victims, servers, and phishers; and examined the frameworks and technologies they utilized.

10 CONCLUSION

In this paper, we comprehensively analyze phishing kits at the script level to achieve fundamental understanding of server-side behaviors, and find real-world implications. To this aim, we collect 4,153 phishing kits and 5.8M index HTML scripts from 15.5M phishing URLs for our analysis. We classify three user interaction patterns of phishing kits. By examining the types of submitted information, interaction trials, and exfiltration methods, we find that the potential information leakage increases in the following order: *single stage (none-real-time)*, *multi-stage (non-real-time)*, and *multi-stage (real-time)* phishing kits. Moreover, we discover 161 real-time phishing kits using real-time technologies, showing the current deployment of advanced admin panels in the wild. We measure the impact of the evasive behaviors and their widespread usages on real-world phishing websites and URLs. By comparing landing pages of kits to phishing sites, our research finds that each kit with the same single landing page script is deployed in an average of 46.92 domains, revealing their practical impact on the phishing ecosystem and chronological trends. Our study can further contribute to constructing advanced phishing detection techniques by offering our findings of web interaction, URL patterns/redirection, and kit-deployment trends.

ACKNOWLEDGMENTS

We thank the anonymous referees for their constructive feedback. We also thank APWG for sharing their valuable phishing dataset. The authors gratefully acknowledge the support of NSF (2210137 and 2335798). This work was supported by Basic Science Research Program through the NRF (NRF-2021R1A6A1A13044830) and ICT Creative Consilience Program through the IITP (No.2022-0-00411, IITP-2024-2020-0-01819, IITP-2023-2021-0-01810) funded by the Korea government. This research was also supported by Science Alliance's StART program and gifts from Google exploreCSR and TensorFlow. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors.

REFERENCES

[1] 1997. *cURL - command line tool and library for transferring data with URLs.* https://curl.se/ (Accessed: 10.02.2023).
[2] 2001. *Drupal - Open Source CMS | Drupal.org.* https://www.drupal.org/ (Accessed: 10.02.2023).
[3] 2001. *PHPMailer/PHPMailer: The classic email sending library for PHP.* https://github.com/PHPMailer/PHPMailer (Accessed: 10.02.2023).
[4] 2002. *Dolibarr Open Source ERP and CRM - Web suite for business.* https://www.dolibarr.org/ (Accessed: 10.02.2023).
[5] 2002. *LiveChat Official Site - Live Chat & Help Desk Solution.* https://www.livechat.com/ (Accessed: 10.02.2023).
[6] 2002. *Moodle - Open-source learning platform | Moodle.org.* https://moodle.org/ (Accessed: 10.02.2023).
[7] 2002. *Redis: The open source, in-memory data store used by millions of developers as a database, cache, streaming engine, and message broker.* https://redis.io/ (Accessed: 10.02.2023).
[8] 2003. *Anti-Phishing Working Group. eCrime Exchange.* https://apwg.org/ecx/ (Accessed: 10.02.2023).
[9] 2003. *Anti-Phishing Working Group. ,Phishing Activity Trends Report.* https://apwg.org/trendsreports/ (Accessed: 10.02.2023).
[10] 2003. *Laravel Echo library for beautiful Pusher and Ably integration.* https://github.com/laravel/echo (Accessed: 10.02.2023).
[11] 2005. *Joomla Content Management System (CMS) - try it! It's free!* https://www.joomla.org/ (Accessed: 10.02.2023).
[12] 2005. *WordPress.com: Fast, Secure Managed WordPress Hosting.* https://wordpress.com// (Accessed: 10.02.2023).
[13] 2006. *Welcome to CodeIgniter.* https://www.codeigniter.com/ (Accessed: 10.02.2023).
[14] 2007. *Google Inc. Safe Browsing - Google Safe Browsing.* https://safebrowsing.google.com/ (Accessed: 10.02.2023).
[15] 2008. *Twilio: Communication APIs for SMS, Voice, Video.* https://www.twilio.com/ (Accessed: 10.02.2023).
[16] 2011. *Bootstrap · The most popular HTML, CSS, and JS library in the world.* https://getbootstrap.com// (Accessed: 10.02.2023).
[17] 2011. *Pusher | Leader In Realtime Technologies.* https://pusher.com/ (Accessed: 10.02.2023).
[18] 2012. *Selenium.* https://www.selenium.dev/ (Accessed: 10.02.2023).
[19] 2012. *Sendinblue.com - More than Just Email Marketing.* https://www.sendinblue.com/ (Accessed: 10.02.2023).
[20] 2015. *Telegram Bot API.* https://core.telegram.org/bots/api (Accessed: 10.02.2023).
[21] 2016. *fake-useragent 0.1.11.* https://pypi.org/project/fake-useragent/ (Accessed: 10.02.2023).
[22] 2016. *gRPC.* https://grpc.io/ (Accessed: 10.02.2023).
[23] 2018. *Mercure.rocks: Real-time APIs Made Easy.* https://mercure.rocks/ (Accessed: 10.02.2023).
[24] 2019. *Laravel - The PHP Framework For Web Artisans.* https://laravel.com// (Accessed: 10.02.2023).
[25] Sahar Abdelnabi, Katharina Krombholz, and Mario Fritz. 2020. VisualPhishNet: Zero-day phishing website detection by visual similarity. In *Proceedings of the 2020 ACM SIGSAC conference on computer and communications security*. 1681–1698.
[26] Bhupendra Acharya and Phani Vadrevu. 2021. {PhishPrint}: evading phishing detection crawlers by prior profiling. In *30th USENIX Security Symposium (USENIX Security 21)*. 3775–3792.
[27] Anupama Aggarwal, Ashwin Rajadesingan, and Ponnurangam Kumaraguru. 2012. PhishAri: Automatic realtime phishing detection on twitter. In *2012 eCrime Researchers Summit*. IEEE, 1–12.
[28] Hugo Bijmans, Tim Booij, Anneke Schwedersky, Aria Nedgabat, and Rolf van Wegberg. 2021. Catching Phishers By Their Bait: Investigating the Dutch Phishing Landscape through Phishing Kit Detection. In *30th USENIX Security Symposium (USENIX Security 21)*. 3757–3774.
[29] Aaron Blum, Brad Wardman, Thamar Solorio, and Gary Warner. 2010. Lexical feature based phishing URL detection using online learning. In *Proceedings of the 3rd ACM Workshop on Artificial Intelligence and Security*. 54–60.
[30] Jiann-Liang Chen, Yi-Wei Ma, and Kuan-Lung Huang. 2020. Intelligent Visual Similarity-Based Phishing Websites Detection. *Symmetry* 12, 10 (2020), 1681.
[31] Sidharth Chhabra, Anupama Aggarwal, Fabricio Benevenuto, and Ponnurangam Kumaraguru. 2011. Phi. sh/$ ocial: the phishing landscape through short urls. In *Proceedings of the 8th Annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference*. 92–101.
[32] Marco Cova, Christopher Kruegel, and Giovanni Vigna. 2008. There Is No Free Phish: An Analysis of" Free" and Live Phishing Kits. *WOOT* 8 (2008), 1–8.
[33] Ian Fette and Alexey Melnikov. 2011. Rfc 6455: The websocket protocol.
[34] Sujata Garera, Niels Provos, Monica Chew, and Aviel D Rubin. 2007. A framework for detection and measurement of phishing attacks. In *Proceedings of the 2007 ACM workshop on Recurring malcode*. 1–8.
[35] Srishti Gupta and Ponnurangam Kumaraguru. 2014. Emerging phishing trends and effectiveness of the anti-phishing landing page. In *2014 APWG Symposium on Electronic Crime Research (eCrime)*. IEEE, 36–47.
[36] Xiao Han, Nizar Kheir, and Davide Balzarotti. 2016. Phisheye: Live monitoring of sandboxed phishing kits. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. 1402–1413.
[37] Kanchan Hans, Laxmi Ahuja, and Sunil Kumar Muttoo. 2017. Detecting redirection spam using multilayer perceptron neural network. *Soft Computing* 21 (2017), 3803–3814.
[38] Grant Ho, Asaf Cidon, Lior Gavish, Marco Schweighauser, Vern Paxson, Stefan Savage, Geoffrey M Voelker, and David Wagner. 2019. Detecting and characterizing lateral phishing at scale. In *28th USENIX Security Symposium (USENIX Security 19)*. 1273–1290.
[39] Microsoft Threat Intelligence. 2021. Catching the big fish: Analyzing a large-scale phishing-as-a-service operation. https://www.microsoft.com/en-us/security/blog/2021/09/21/catching-the-big-fish-analyzing-a-large-scale-phishing-as-a-service-operation/ (Accessed: 10.02.2023).
[40] Luca Invernizzi, Kurt Thomas, Alexandros Kapravelos, Oxana Comanescu, Jean-Michel Picod, and Elie Bursztein. 2016. Cloak of visibility: Detecting when machines browse a different web. In *2016 IEEE Symposium on Security and Privacy (SP)*. IEEE, 743–758.
[41] Brian Kondracki, Babak Amin Azad, Oleksii Starov, and Nick Nikiforakis. 2021. Catching Transparent Phish: Analyzing and Detecting MITM Phishing Toolkits. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. 36–50.
[42] Martijn Koster, Gary Illyes, Henner Zeller, and Lizzi Sassman. 2022. Robots Exclusion Protocol. RFC 9309. https://doi.org/10.17487/RFC9309
[43] Luda Lazar. 2018. Our Analysis of 1,019 Phishing Kits. https://www.imperva.com/blog/our-analysis-of-1019-phishing-kits/
[44] Yun Lin, Ruofan Liu, Dinil Mon Divakaran, Jun Yang Ng, Qing Zhou Chan, Yiwen Lu, Yuxuan Si, Fan Zhang, and Jin Song Dong. 2021. Phishpedia: A Hybrid Deep Learning Based Approach to Visually Identify Phishing Webpages.. In *USENIX Security Symposium*. 3793–3810.
[45] Ruofan Liu, Yun Lin, Yifan Zhang, Penn Han Lee, and Jin Song Dong. 2023. Knowledge Expansion and Counterfactual Interaction for {Reference-Based} Phishing Detection. In *32nd USENIX Security Symposium (USENIX Security 23)*. 4139–4156.
[46] Wenyin Liu, Xiaotie Deng, Guanglin Huang, and Anthony Y Fu. 2006. An antiphishing strategy based on visual similarity assessment. *IEEE Internet Computing* 10, 2 (2006), 58–65.
[47] Imran Khan Marcel Feller, Fahad Iqbal. 2021. We're gonna need a bigger boat: An analysis of recently caught phishing kits. https://www.nortonlifelock.com/blogs/norton-labs/phishing-kits
[48] Samuel Marchal, Kalle Saari, Nidhi Singh, and N Asokan. 2016. Know your phish: Novel techniques for detecting phishing sites and their targets. In *2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 323–333.
[49] Sourena Maroofi, Maciej Korczyński, and Andrzej Duda. 2020. Are you human? resilience of phishing detection to evasion techniques based on human verification. In *Proceedings of the ACM Internet Measurement Conference*. 78–86.
[50] Ettore Merlo, Mathieu Margier, Guy-Vincent Jourdan, and Iosif-Viorel Onut. 2022. Phishing Kits Source Code Similarity Distribution: A Case Study. In *2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*. IEEE, 983–994.
[51] Adam Oest, Yeganeh Safaei, Adam Doupé, Gail-Joon Ahn, Brad Wardman, and Kevin Tyers. 2019. Phishfarm: A scalable framework for measuring the effectiveness of evasion techniques against browser phishing blacklists. In *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1344–1361.
[52] Adam Oest, Yeganeh Safaei, Penghui Zhang, Brad Wardman, Kevin Tyers, Yan Shoshitaishvili, and Adam Doupé. 2020. {PhishTime}: Continuous Longitudinal Measurement of the Effectiveness of Anti-phishing Blacklists. In *29th USENIX*

Security Symposium (USENIX Security 20). 379–396.

[53] Adam Oest, Yeganeh Safei, Adam Doupé, Gail-Joon Ahn, Brad Wardman, and Gary Warner. 2018. Inside a phisher's mind: Understanding the anti-phishing ecosystem through phishing kit analysis. In *2018 APWG Symposium on Electronic Crime Research (eCrime)*. IEEE, 1–12.

[54] Adam Oest, Penghui Zhang, Brad Wardman, Eric Nunes, Jakub Burgis, Ali Zand, Kurt Thomas, Adam Doupé, and Gail-Joon Ahn. 2020. 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)*.

[55] Peng Peng, Chao Xu, Luke Quinn, Hang Hu, Bimal Viswanath, and Gang Wang. 2019. What happens after you leak your password: Understanding credential sharing on phishing sites. In *Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security*. 181–192.

[56] Tiago Pereira. 2023. New phishing-as-a-service tool "Greatness" already seen in the wild. https://blog.talosintelligence.com/new-phishing-as-a-service-tool-greatness-already-seen-in-the-wild/

[57] Routhu Srinivasa Rao and Alwyn Roshan Pais. 2020. Two level filtering mechanism to detect phishing sites using lightweight visual similarity approach. *Journal of Ambient Intelligence and Humanized Computing* 11, 9 (2020), 3853–3872.

[58] Angelo PE Rosiello, Engin Kirda, Fabrizio Ferrandi, et al. 2007. A layout-similarity-based approach for detecting phishing pages. In *2007 third international conference on security and privacy in communications networks and the workshops-securecomm 2007*. IEEE, 454–463.

[59] Peter Saint-Andre. 2011. RFC 6120: extensible messaging and presence protocol (XMPP): core.

[60] Brett Stone-Gross, Thorsten Holz, Gianluca Stringhini, and Giovanni Vigna. 2011. The Underground Economy of Spam: A Botmaster's Perspective of Coordinating {Large-Scale} Spam Campaigns. In *4th USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET 11)*.

[61] Karthika Subramani, William Melicher, Oleksii Starov, Phani Vadrevu, and Roberto Perdisci. 2022. PhishInPatterns: measuring elicited user interactions at scale on phishing websites. In *Proceedings of the 22nd ACM Internet Measurement Conference*. 589–604.

[62] Bhaskar Tejaswi, Nayanamana Samarasinghe, Sajjad Pourali, Mohammad Mannan, and Amr Youssef. 2022. Leaky Kits: The Increased Risk of Data Exposure from Phishing Kits. In *2022 APWG Symposium on Electronic Crime Research (eCrime)*. 1–13. https://doi.org/10.1109/eCrime57793.2022.10142092

[63] Kurt Thomas, Frank Li, Ali Zand, Jacob Barrett, Juri Ranieri, Luca Invernizzi, Yarik Markov, Oxana Comanescu, Vijay Eranti, Angelika Moscicki, et al. 2017. Data breaches, phishing, or malware? Understanding the risks of stolen credentials. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*. 1421–1434.

[64] Kurt Thomas, Damon McCoy, Chris Grier, Alek Kolcz, and Vern Paxson. 2013. {Trafficking} Fraudulent Accounts: The Role of the Underground Market in Twitter Spam and Abuse. In *22nd USENIX Security Symposium (USENIX Security 13)*. 195–210.

[65] Ke Tian, Steve TK Jan, Hang Hu, Danfeng Yao, and Gang Wang. 2018. Needle in a haystack: Tracking down elite phishing domains in the wild. In *Proceedings of the Internet Measurement Conference 2018*. 429–442.

[66] Huahong Tu, Adam Doupé, Ziming Zhao, and Gail-Joon Ahn. 2019. Users really do answer telephone scams. In *28th USENIX Security Symposium (USENIX Security 19)*. 1327–1340.

[67] Susana Vinga and Jonas Almeida. 2003. Alignment-free sequence comparison—a review. *Bioinformatics* 19, 4 (2003), 513–523.

[68] Suzanne Widup, Alex Pinto, David Hylender, Gabriel Bassett, and Philippe langlois. 2021. 2021 Verizon Data Breach Investigations Report.

[69] Ammara Zamir, Hikmat Ullah Khan, Tassawar Iqbal, Nazish Yousaf, Farah Aslam, Almas Anjum, and Maryam Hamdani. 2020. Phishing web site detection using diverse machine learning algorithms. *The Electronic Library* 38, 1 (2020), 65–80.

[70] Penghui Zhang, Adam Oest, Haehyun Cho, Zhibo Sun, RC Johnson, Brad Wardman, Shaown Sarker, Alexandros Kapravelos, Tiffany Bao, Ruoyu Wang, et al. 2021. Crawlphish: Large-scale analysis of client-side cloaking techniques in phishing. In *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1109–1124.

[71] Penghui Zhang, Zhibo Sun, Sukwha Kyung, Hans Walter Behrens, Zion Leonahenahe Basque, Haehyun Cho, Adam Oest, Ruoyu Wang, Tiffany Bao, Yan Shoshitaishvili, et al. 2022. I'm SPARTACUS, No, I'm SPARTACUS: Proactively Protecting Users from Phishing by Intentionally Triggering Cloaking Behavior. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. 3165–3179.

APPENDIX

A AUTOMATIC & MANUAL ANALYSIS OF USER INTERACTION

Automatic Analysis. As explained in Section 5.1, HTML or PHP scripts, which show phishing contents such as fake login pages, contain navigating functions that move victims' accesses to the next scripts. For example, if a victim inputs ID and password on the login page, the functions navigate the access to the input error page, making the victim inputs ID and password once again.

We follow the destination scripts of the navigation functions (`window.location` JavaScript functions and `header` PHP functions). *First,* we search the navigation functions in the main page script which are usually located in the root directory of phishing kit with the name, 'index.php' or 'index.html'. *Second,* we check if a destination script of a navigation function exists. If the destination script exists, we make a navigation tree whose root node is the script where the navigation function exists and leaf node is the destination script. *Third,* we repeat this process until navigation functions are not found in any destination scripts. As a result, the height of the navigation tree represents the number of user interactions.

Manual Analysis. At the same time, if following the navigation fails (*e.g.*, none of user interactions are found), we manually analyze those kits, and check whether a specific framework with various plugins constitutes the kit and uses its navigation functions (*e.g.*, `redirect` function for CodeIgniter [13]) found by our automatic analysis of navigation because of obfuscation.

We observed 161 of all 204 failures because of using their own navigation functions and specific routing scripts. The other 43 failed kits have obfuscated their navigation functions such as using base64 encoded function names. We double-checked their user interactions by manually translating their obfuscated functions and deploying the kits in the local server, and confirm that all 43 kits are ❷ non-real-time multi-stage kits.

B DETAIL OF MULTI-TARGETING BRAND PANELS

As shown in Table A2, we check the targeted brands of these phishing kits using the same methodology described in Section 5.2.2. Specifically, `XBALTI` mainly targets Chase and Amazon, while the other three identities target various banks. Interestingly, one of `Spox` panels provides twelve targeting bank brands to phishing attackers, while most of `XBALTI` only provide two targeting brands, Chase and Amazon. We suspect that the difference may happen because phishing kit creators have sold different phishing kits depending on the price paid by phishing attackers. For example, we can choose what brand to target on the admin panel or the configuration script, and the creator provides targeting brand plugins if we pay more. These multi-targeting brand panels work like as small phishing-as-a-services [39, 56], which make it even more difficult to identify the subject of phishing campaigns. They facilitate templates of various brands, 2FA bypassing, evasion techniques, and Telegram bots, all in one to unskilled kit users, increasing potential phishing attackers.

C USAGE OF BOGUS PARAMETERS

As shown in Listing A4 in Appendix, line 3 just checks if the URL before redirecting to this script has the string of bogus HTTP GET parameter, '_Authentication', and value, 'Service_Login_', written and do nothing if the string is in URL. In contrast, all HTTP GET values of the remaining kits pertain to either ID or email addresses, which they use as filling welcome phrases on their fraudulent login pages. This implies that the purpose of the bogus parameters is simply to bypass URL-based blocklists, rather than providing meaningful information along with the URLs. Moreover, as shown in Table A5, well-known services use HTTP GET Value patterns for their login pages. Comparing whether the parameter is requested by HTTP GET can be an important criterion for determining a phishing URL.

D LISTINGS

Listing A1: An Example of a Configuration Script of *uadmin* Phishing Kit Family.

```
<?php
...
// $php_js->home=$relative_root."email.php";
// $php_js->home=$relative_root."jabb.php";
$php_js->home=$relative_root."home.php";
// $php_js->home = "http://localhost/uadmin/gate.php?pl=
    logs";
// $php_js->stat_home="http://127.0.0.1/uadmin/gate.php";
// $php_js->home="http://127.0.0.1/uadmin/gate.php?pl=
    token";
...
?>
```

Listing A2: Example of Script Dealing With Information Exfiltration in the Phishing Kit

```
<?php
@$message .= "---------------|V0y493|---------------\n";
$message .= "USER AGENT: ".$browser."\n\n";
$message .= "Email: " . $_POST['login'] . "\n";
$message .= "Password: " . $_POST['passwd'] . "\n\n";
$message .= "IP : " .$ip. "\n";
$message .= "
    ------------------------------------------\n";
?>
```

Listing A3: Dynamically-generated URLs employed in the Phishing Kit

```
[index.php]
<?php
include function.php
...
$name =  generateRandomString();
$secfile = $name.".php";
if (!copy($staticfile, $secfile)) {
//echo "file not create\n";
}else {
if(file_exists($secfile)){
//echo "file exist\n";
unlink($staticfile);
header("Location: $secfile?rand=13InboxLightaspxn
    .1774256418&fid&1252899642&fid.1&fav.1");
}}

...
?>

[function.php]
<?php
```

```
...
$name =  generateRandomString();
function generateRandomString($length = 24) {
    $characters = '0123456789abcdefghijklmnopqrstuvwxyz';
    $charactersLength = strlen($characters);
    $randomString = '';
    for ($i = 0; $i < $length; $i++) {
        $randomString .= $characters[rand(0,
            $charactersLength - 1)];
    }
    return $randomString;
}
?>
```

Listing A4: Fake HTTP GET parmeters Creation in Kits

```
<?php
...
if (strpos($actual_link, 'Service_Login_&_Authentication'
    ) !== false) {
    //Do nothing
}elseif(strpos($actual_link, '?') != true){
    $url = $actual_link."?&SERVID=Service_Login_&
        _Authentication=".sha1(uniqid(time()));
    header("Location: ".$url);
}else{
    $url = $actual_link."&SERVID=Service_Login_&
        _Authentication=".sha1(uniqid(time()));
    header("Location: ".$url);
}
...
?>
```

E TABLES

Table A1: Top 10 Targeting Brands for each Type of User Interaction Pattern

Type	Brand	#Kits	Type	Brand	#Kits
❶Single stage Phishing	Microsoft	444	❸Multi-stage real-time	Microsoft	47
	Adobe	74		Wells Fargo	15
	Facebook	47		DHL	12
	Wells Fargo	44		PayPal	8
	TMOBILE	38		Chase	5
	AT&T	29		Facebook	4
	Fake Email	22		Societe Generale	4
	Yahoo	19		Commonwealth Bank	4
	Chase	19		Adobe	3
	DHL	19		Orange	3
❷Multi-stage non-real-time	Microsoft	779	**Total**	Microsoft	1,270
	Wells Fargo	176		Wells Fargo	235
	Chase	174		Chase	198
	DHL	148		DHL	179
	Fake Email	96		Adobe	159
	Facebook	83		Facebook	134
	USPS	83		Fake Email	120
	Adobe	82		USPS	86
	Netflix	68		Netflix	74
	Amazon	49		PayPal	71

Table A2: Phishing Kits Supporting Multi-brand Panels

Kit	User Interaction Pattern	Targeted Brands (#Kits)	#Kits
XBALTI	❷Multi-stage non-real-time	Chase: 49, Amazon: 14, Others 24: 48	111
Spox	❷Multi-stage non-real-time	Wells Fargo: 15, Chase: 14, Others 10: 18	47
uAdmin	❸Multi-stage real-time	DHL: 6, CB*: 4, Others 13: 20	30
NewOreoo	❷Multi-stage non-real-time	SB**: 9, Chase: 7, Others 7: 10	26

*:Commonwealth Bank, **: Standard Bank

Table A3: Redirection Functions from Index Scripts of Phishing Kits

Lang.	Functions		#Scripts (%)	Avg. Usages*	Max. Usages*
PHP	header		3,883 (65.52%)	1.248	47
HTML	meta tag		717 (12.09%)	0.068	4
JavaScript	window.location	.href	717 (12.09%)	0.187	15
		.replace	728 (12.28%)	0.255	5
		.assign	23 (0.39%)	0.004	3
		.pathname	23 (0.39%)	0.004	3
		.reload	34 (5.74%)	0.205	2

*: Usages per Script

Table A4: 20 Types of Information Exfiltrated from Kits

ID & Mail Add.	*Password*	*Phone*	*Address*
ZIP code	*Route Number*	*Card Number*	*CVV*
ExpDate	*Question*	*Answer*	*Name*
OTP	*Birth Date*	*Account Number & Password*	*Driver License*
SSN	*ATM PIN*	*Crypto Wallet Phrase & Key*	*Verifying code*

Table A5: Authentic URL Patterns of top 30 Brands from Phishing URLs

Brand Name	URL pattern	#Brands	Example
Microsoft, Facebook, Instagram, WellsFargo, DHL, Adobe, USPS, Yahoo, Amazon, AT&T, CommonWealth, Standard Bank, Banco de Chile, cpanel, Twitter(X), Santander	HTTP GET Value	16	https://id.cpanel.net/get/login?url=aHR0cH~(cpanel)
Chase, Netflix, Paypal, TMobile, Societe Generale Group, ING Group, Apple, Credit Agricole	Name of Directories	8	https://secure.chase.com/web/auth/#/logon/logon/chaseOnline (Chase)
Absa, Dibaserver, Sparkasse, winbank, BancoEstado, ABN AMRO	Name of Scripts	6	https://ib.absa.co.za/absa-online/login.jsp (Absa)

Table A6: Kit Deployment by User Interaction Types

Brand	Landing Page	# Kits	# Domains	Brand	Landing Page	#Kits	#Domains	Brand	Landing Page	# Kits	# Domains
Microsoft	Microsoft 1	84	3,393	Facebook	Facebook 1	2	3,881	MAG*	MAG 1	172	6,939
	Microsoft 2	83	3,185		Facebook 2	3	1,345		MAG 2	1	192
	Microsoft 3	3	2,850		Facebook 3	3	579		MAG 3	1	178
	Others: 143	472	13,085		Others: 27	51	4,299				
	Total: 146	642	22,513		Total: 30	59	10,104		Total: 3	174	7,309
Chase	Chase 1	5	3,353	Wells Fargo	WF 1	22	856	Amazon	Amazon 1	2	1,690
	Chase 2	24	1,958		WF 2	8	675		Amazon 2	1	712
	Chase 3	2	323		WF 3	5	628		Amazon 3	6	588
	Others: 17	32	1,144		Others: 26	108	4,029		Others: 8	8	129
	Total: 20	63	6,778		Total: 29	143	6,188		Total: 11	17	3,119

*: **Microsoft - Adobe Generic.** The brand displays a fake Adobe login page where the potential victim logins with a Microsoft account.