Uncovering the Role of Support Infrastructure in Clickbait PDF Campaigns

Giada Stivala*, Gianluca De Stefano*, Andrea Mengascini*, Mariano Graziano†, Giancarlo Pellegrino*
* *CISPA Helmholtz Center for Information Security*
Saarbrücken, Germany
{*giada.stivala, gianluca.de-stefano, andrea.mengascini, pellegrino*}*@cispa.de*
†*Cisco Talos*
London, UK
graziano.mariano@gmail.com

Abstract—**Clickbait PDFs, an entry point for multiple Web attacks, are distributed via SEO poisoning and rank high in search results due to being massively uploaded on abused or compromised websites. The central role of these hosts in the distribution of clickbait PDFs remains understudied, and it is unclear whether attackers differentiate the types of hosting for PDF uploads, how long they rely on hosts, and how affected parties respond to abuse.**

To address this, we conducted real-time analyses on hosts, collecting data on 4,648,939 clickbait PDFs served by 177,835 hosts over 17 months. Our results revealed a diverse infrastructure, with hosts falling into three main hosting types. We also identified at scale the presence of eight software components which facilitate file uploads and which are likely exploited for clickbait PDF distribution. We contact affected parties to report the misuse of their resources via a large-scale vulnerability notification. While we observed some effectiveness in terms of number of cleaned-up PDFs following the notification, long-term improvement in this infrastructure remained insignificant. This finding raises questions about the hosting providers' role in combating abuse and the actual impact of vulnerability notifications.

1. Introduction

Phishing and spam have been known for decades and, nonetheless, they keep being a profitable option for cybercriminals [?]. While most known for spreading via e-mail [?], mediums for these attacks have evolved in time, encompassing a variety of means, such as SMS, phone calls, social media platforms [? ?], and the more recent clickbait PDFs [?]. Clickbait PDFs are PDF documents whose first page contains a visual bait embedding a link to a Web attack such as phishing attacks [?], malware download [?], and malicious browser extension download [?]. They are distributed via search engine (SE) poisoning attacks, ranking high as these files are hosted in benign servers and cross-reference one another, increasing their page rank. Only recently the research community has started looking into this new threat, focusing on visual baits [?], type of Web attack [? ?], and volumetric features [? ?], but neglecting the role played by the supporting infrastructure in these attacks.

The effectiveness of clickbait PDF attacks relies on massive daily uploads of cross-linked PDFs in benign servers [?]. Studying the supporting infrastructure has been a critical aspect when analyzing other similar threats, such as drive-by download [?], phishing pages [?], spam [?], or when looking at server compromise [? ?] and their role in the attacks [?]. Despite previous research efforts, there are still gaps in our understanding of the supporting infrastructure. These findings do not directly apply to clickbait PDFs, as they focus on threats with different core features or examine the infrastructure with limited scope. Firstly, it remains unclear which and how many types of hosting are utilized to serve clickbait PDFs. Prior works, when considering the type of hosting in their analyses, have used it as a pre-filtering criterion (e.g., only Cloud Storage [?]) or focused on a predefined list of domains [?]. Another key feature of this threat is the extensive volume of PDFs online on infrastructure hosts for prolonged periods, ensuring their presence in poisoned search results [?]. This temporal aspect differs among malicious actions where, for example, phishing pages stay online for 1-2 days [? ?], malware components for maximum 5.5 days [?] and scripts for SEO pages for a maximum of 30 days [?]. The duration of the uptime of malicious resources is an attack-specific characteristic, and does not transfer straightforwardly to the threat posed by clickbait PDFs. In fact, we lack knowledge about the duration of PDFs remaining online on these hosts and how website owners and hosting providers respond to such abuse of their resources. Finally, it is unclear whether attackers rely on compromised websites as support hosts and which software component they exploited for the upload of clickbait PDFs. While previous works showed that it is possible to find compromised websites by starting from vulnerable components [? ?], we take the opposite approach and empirically enumerate the different exploits attackers might have used to upload clickbait PDFs on the hosts supporting the attack.

This paper sheds light on and provides a comprehensive description of the infrastructure behind clickbait PDF attacks. Employing a data-driven approach, we conduct a large-scale study of website abuse aimed at distributing clickbait PDFs. Starting from a feed of real-world PDFs, we identify clickbait PDFs in this feed and leverage their cross-link structure to identify a large portion of the supporting infrastructure, which we further study with an array of specific analyses. Our measurements and observations quantify the volume of hosts involved in this phenomenon and reveal the impact on various hosting types. Additionally, we identify three affected hosting types (*Object storage*, *Website hosting*, and *CDN*), demonstrating that the clickbait PDFs threat spans across multiple hosting categories. Additionally, this paper investigates the factors exploited by attackers to gain access to these hosts. We collect metadata on software components running at the origins in our dataset and possible indicators of compromise with analyses tailored to the specific type of hosting. Our findings reveal a fragmented picture, where attackers leverage characteristics specific to the type of hosting, or provider, to gain access to hosting space. For example, we identified eight software components that facilitate file uploads, along with 12,927 origins running outdated software.

This paper also investigates ways to help mitigate the distribution of clickbait PDFs, whose threat is ongoing since 2020 [?]. We identify hosting providers as entities affected by this malicious action, whose resources are abused to serve clickbait PDFs to victim users, and reach out to them to seek their cooperation in fighting this abuse. We undertake a vulnerability notification procedure to limit the distribution of clickbait PDFs, raise awareness of this threat and collect any feedback from affected parties. Our observations show statistically significant results in the cleanup of phishing PDFs, with an overall positive feedback from the notified parties. Worryingly, the benefits of this action appear not to last long. Notified websites serve previously unseen clickbait PDFs in 97% of the cases, indicating that most providers do not deal with the originating causes allowing file uploads.

In summary, our paper makes the following contributions:

- We present a comprehensive picture of the infrastructure supporting clickbait PDF attacks, regardless of the hosting type or provider, and empirically define the volume and the duration of the abuse.
- We identify three types of hosting the websites in our dataset belong to via a systematic methodology and a thorough manual validation.
- We identify outdated and vulnerable software components, likely connected to host exploitation and to file upload.
- We help mitigate the spread of clickbait PDFs by informing 1,545 affected parties hosting 799,930 PDFs about their presence.

2. Background and Research Questions

2.1. Background

In this section, we introduce core concepts, outline the framing of our study and present our research questions.

2.1.1. Clickbait PDFs. Clickbait PDFs were recently presented in [?] as fraudulent-looking PDFs functioning as an entry point for a series of Web attacks (phishing, drive-by download, scam, etc), should the victim have clicked on a link embedded in their first page. In this attack scenario, victim users come across clickbait PDFs when searching for specific terms on search engines (such as Google and Bing). The PDFs are returned among the first ten search results and seamlessly rendered by the browser upon a click [?]. This high rank in search results is attributed to malicious search engine optimization techniques, which exploit the structure and content of the PDFs to achieve higher rankings. Specifically, clickbait PDFs display a bait message or image in the first page, which also embeds the link leading to a Web attack, while including 13 to 30 links to different clickbait PDFs ([?]) in the following pages to implement backlinking and resources cross-linking to boost SE ranking ([? ?]). Since SEO attacks are the main distribution vector for clickbait PDFs [?], the rest of this paper focuses on clickbait PDFs distributed through SEO. **??** displays these interconnections between different clickbait PDFs.

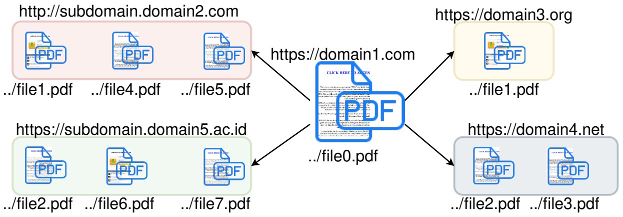


Figure 1: The interconnections between clickbait PDFs.

2.1.2. SEO Attacks. Search engine optimization (SEO) attacks aim at promoting malicious content high in search results. Previous studies established a series of techniques used by attackers in a SEO attack, such as *keyword stuffing* [?], where the promoted content is filled with specific key terms to appear more relevant. *Cross-linking resources* [?] exploits the link-based ranking algorithms of search engines, where attackers craft a network of ad-hoc resources and cross-link them to influence the ranking of promoted resources. Finally, attackers *use benign websites* to host the cross-linked resources [?] as their good reputation positively influences the final ranking.

2.1.3. Websites Supporting Web Attacks. Many works discussing URL maliciousness consider a website serving malicious content as either being owned by the attacker or compromised [? ? ?]. However, this perspective overlooks scenarios where the attacker neither registers a new domain nor compromises a third party's domain, but rather get assigned a domain from a hosting provider (e.g., a free subdomain). This possibility became feasible with the availability of inexpensive (if not free) services offered by hosting providers, possibly without thorough registration checks. These services may include free object storage, free subdomains for E-commerce websites, or online marketplaces. When such infrastructure is used for malicious purposes, it is technically incorrect to call it "compromised" since the attacker did not exploit the software stack running at that origin. We thus use the broader term *abused infrastructure* to indicate a large amount of websites, potentially managed by a single provider or part of the same hosting service, whose usage is inappropriate, often illicit, resulting in significant harm to the owner and its users. In the context of clickbait PDFs, the supporting infrastructure is the ensemble of websites, services and providers whose resources are being misused by attackers to host clickbait PDFs, such as `domain1.com`, `subdomain.domain2.com`, `domain3.org` and `domain4.net` in **??**.

2.2. Scope and Contributions

In this section, we first present our research questions, then, we outline the contributions and framing of this study with respect to a recent work in this field.

2.2.1. Research Questions. The overarching goal of this study is to observe the web infrastructure abused for the distribution of clickbait PDFs, investigating specific properties concerning its volume and evolution in time. The first challenge we undertake (**Research Question 1**) is to understand its composition in terms of hosts or services, for example by identifying Autonomous Systems or any specific hosting services involved, and to which extent. We tackle this research question in § **??**. Next, we ask ourselves how attackers acquire upload capabilities to these domains (**Research Question 2**). Specifically, we look for security-related properties, as the presence of outdated, vulnerable or misconfigured software components which might have been exploited by attackers to gain the ability of uploading clickbait PDFs. We investigate multiple security properties and report our findings in § **??**. Following, we focus on the duration and volume of the abuse (**Research Question 3**). We define the duration of abuse by monitoring the online status of all clickbait PDFs in our dataset with the granularity of a single day (§ **??**), and its volume by observing the distribution of clickbait PDFs over the types of hosting we previously identified (§ **??**). Lastly, we focus on measures that could be taken to help mitigating the spread against clickbait PDFs, ultimately protecting users and improving the security of the abused hosts. Existing protection methods, as blocklists, provide limited protection for users (§ **??**), thus, we evaluate the effectiveness of responsibly disclosing the issue to affected parties (**Research Question 4**) (§ **??**), observing as impact indicators both the number of PDFs that were cleaned up and the domains that did (or did not) see any further upload.

2.2.2. Contributions. The closest work to ours is a recent study by Stivala et al. [**?**]. In the following, we elaborate on the differences between the two works, outlining our contributions. Our work expands on the findings in [**?**] by shifting the focus to the *abused infrastructure* hosting clickbait PDFs. Our investigation centers on identifying hosting types (RQ1), gathering evidence of upload methods (RQ2), and the impact of responsible disclosure (RQ4), differently from [**?**] which focuses on PDF characteristics and distribution methods. To enhance our understanding of real-time abuse monitoring (RQ3), we introduce two datasets: *Seed DS* and *Main DS*, where *Seed DS* serves as source to build *Main DS*, and *Main DS* allows for direct real-time analysis. Our *Seed DS* is newer and three times larger than [**?**]'s dataset, with no temporal overlap nor shared samples. When assessing the volume and duration of this phenomenon, a shared point of investigation, we focus on live hosts rather than on PDFs: the `.pdf` links in the *Seed DS* enable direct, real-time abuse monitoring–an aspect not studied in [**?**] and significantly different from observing VT uploads [**?**]. Finally, as a technical improvement, we created a new ML model for clustering, reducing latency and human bias.

	DS	Setup Phase	Main Study
Start		2022-03-14	2022-06-22
End		2022-06-21	2023-07-26
PDFs	□	105,598	503,978
of which SEO	□	66,614	384,601
Extracted `.pdf` links	-	1,350,201	4,648,939
of which online & SEO	■	-	2,710,959

TABLE 1: Volume of unique PDFs in *Seed DS* (□) and *Main DS* (■), and unique `.pdf` links extracted from them.

3. Dataset and Pipeline

3.1. Main and Seed Datasets

Answering our research questions requires knowledge of the hosts serving clickbait PDFs, for example in the form of a list of URLs leading to these PDFs. A source of URLs is given by clickbait PDFs themselves, as clickbait PDFs include URLs to other clickbait PDFs as backlinks (see § **??**, **??**, and **??**). We leverage this property to construct a first dataset of clickbait PDFs, the *Seed DS*, acting as source of URLs to other clickbait PDFs. By visiting these URLs and downloading the corresponding PDFs we build the dataset for this study, *Main DS*. The inclusion of a downloaded PDF to the *Main DS* (as well as to the *Seed DS* in the previous step) is subject to the evaluation of SEO-specific properties (detailed in § **??** below), ensuring that no benign or non-clickbait PDF is included.

3.1.1. Data Collection. Our starting dataset, *Seed DS*, counts 609,576 PDFs with unique SHA-256 signatures, covering a period of 17 months (from March 14th, 2022 to June 26th, 2023). The nine-month gap between the start of our study and the end of Stivala et al.'s raises questions about whether those clickbait PDFs are still online and part of an attack campaign, which we address by collecting up-to-date clickbait PDFs provided by two industrial partners, who retrieve them from VirusTotal. The two partners contribute unevenly, accounting for 69% and 29% of the entire dataset, respectively.

We start downloading PDFs to construct the *Main DS* after a three-month setup phase. This second data collection lasted 13 months, during which we monitored 4,648,939 `.pdf` links. URLs that are unreachable, do not serve PDFs, or serve non-clickbait PDFs are discarded, resulting in 2,710,959 URLs that returned a clickbait PDF at least once during the Main phase of the study. **??** reports the number of clickbait PDFs in the *Seed DS* and the number links extracted from them, as well as the number of clickbait PDF observed online. § **??** reports the implementation steps behind our data collection.

3.1.2. Filtering Criteria. We implement two filtering criteria to limit the inclusion of benign or non-clickbait PDFs in our datasets, in line with prior works [**?**]. Identifying clickbait PDF involves verifying the presence of SEO characteristics, which are not visible from the `.pdf` URLs but can be observed by inspecting the PDF structure and content (see § **??**). We thus download and parse PDFs, ensuring the presence of SEO characteristics in two ways before adding them to the *Seed DS* and *Main DS*. These

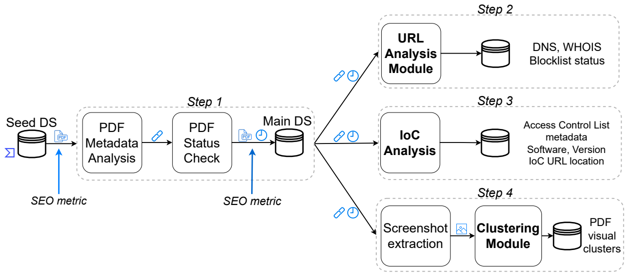


Figure 2: *Grape* modules and I/O data connections.

criteria (hereinafter *SEO metric*) ensure the presence of at least five `.pdf` links in total, relaxed from the original ten, and a mean number of at least one `.pdf` link per page, consistent with [**?**]. This change was due to our different data sources (VirusTotal and backlinks in clickbait PDFs) where the distribution of benign documents is much lower than that of search engines like Google and Bing. The lower threshold is designed to include a large number of clickbait PDF documents while minimizing false positives. § **??** reports on the accuracy of this metric.

3.2. The *Grape* Pipeline

The initial three-month setup phase are necessary to build *Grape*, shown in **??**, a modular pipeline running daily in real-time. *Grape* ingests and processes millions of tiny PDF-related pieces of information from various sources every day. When "mashed" together, these pieces reveal valuable insights into the clickbait PDF threat. We release the code of *Grape* at https://github.com/emerald1010/hosts-supporting-clickbait-PDFs.

The first module (*Step 1*) processes the PDF binaries received from our industry partners, extracting useful metadata such as the embedded URLs. These are fed into the *PDF Status Check*, which visits them and defines their online or offline status. These pairs `(URL, datetime_information)` constitute the basis of the *Main DS* and are the input of all following modules. Specifically, we fetch DNS and WHOIS of all URLs in the *Main DS* (*Step 2*), visit the websites hosting online PDFs looking for indicators of compromise (*Step 3*) and, finally, download online PDFs and extract the screenshot of the first page (*Step 4*) to determine the groups of visual baits. The modules are orchestrated and monitored via an instance of Apache Airflow. Following, we detail the behavior of each module.

3.2.1. PDF Analysis Module.

We begin by choosing PDFs from the *Seed DS* that meet the SEO metric. Next, we extract their URLs and metadata (*PDF Metadata Analysis*), and subsequently verify the online status of those URLs leading to a PDF (*PDF Status Check*). In the *PDF Metadata Analysis*, the URLs are obtained by reconstructing the PDF tree with a modified version of the open-source library `peepdf` [**?**], by navigating the tree breadth-first looking for nodes encoding URLs (e.g., `URI`) or whose parent node's attributes include `Subtype Link`, `Rect` and either `Type Annot` or `Type A`. This approach was preferred to a simple string matching (e.g., looking for `http://`-like strings) as it allows extracting URLs in compressed streams. Lastly, we collect the document title by inspecting its `Document Information Dictionary` and obtain the screenshot of the first page via the `Poppler` [**?**] utility using 150 dots per inch.

PDF Status Check consists of a module performing daily HTTP requests to the extracted `.pdf` links, defacto recording the uptime of each linked PDF. We monitor each link on a daily basis starting from the day of its initial observation, and continue until it remains offline for three consecutive days. A URL is considered offline when its `Content-Type` header is different from `application/pdf`, or if it returns a status code $>=$ 300. To reduce the load on the target domains, we initially perform `HEAD` requests, and proceed with a `GET` only if the above criteria are met. Moreover, we store the linked clickbait PDF on the first visit. We also included the use of numerous VPN endpoints to check that a given domain is not blocklisting us before marking its URLs as offline. *PDF Status Check* became operative on June 22nd, 2022, marking the start of the Main phase of our study (no PDF was downloaded prior to this date). § **??** discusses possible limitations of this approach and § **??** discusses the measures we took to reduce the load of our analyses on target websites. Before adding new PDFs into the *Main DS*, we ensure they meet the *SEO metric*, and then we reapply the *PDF Metadata Analysis*.

3.2.2. URL Analysis Module.

In this step we perform analyses on the extracted URLs. We collect DNS records of each fully-qualified domain name (FQDN) actively serving clickbait PDFs, extract its IP and fetch the corresponding WHOIS record, including Autonomous System numbers. Next, we collect the blocklist status of each extracted `.pdf` link, using Google SafeBrowsing (preinstalled on more than 84% of users' browsers [**?**]) and VirusTotal, popular both in research and in industry (see, e.g., [**? ?**]) as reference.

3.2.3. Indicators of Compromise Collection Module.

The collection of indicators of compromise is a multifaceted procedure which comprises different analyses depending on the target host. It is performed by two submodules collecting evidence of vulnerable or misconfigured software components.

The first module collects indicators linked to the presence of software components and plugins running on the server-side by visiting with a full-fledged Chrome browser the homepage of a domain actively serving clickbait PDFs. When loading the page, the browser waits up to 15 seconds, intercepting all network requests happening in the background. This functionality is similar to that realized by [**?**], which we incorporate for easier interaction with the Linux Traffic Interface. We then process the network traces applying a rule-based approach (we integrate that of [**?**] for simplicity, similarly to [**?**]) to obtain information on the web server (e.g. Apache), programming languages (e.g., PHP), hosting panels (e.g., Plesk), web application framework (e.g., Wordpress) and add-ons (as WordPress Themes and Plugins).

Our second module is a custom vulnerability scanner developed to verify the presence of misconfigured or vulnerable components which may lead to file upload. The scanner visits pre-selected URL paths which we observe are indicators signalling the presence of a component allowing file upload. § **??** details the inner workings of this component. In case no evidence could be collected we trigger additional analyses for this FQDN, where the Chrome browser visits $n \leq 20$ random pages extracted from the homepage of the domain to possibly observe additional software components.

3.2.4. Clustering Module. Clickbait PDFs can be clustered with respect to the visual deceit (e.g., position and aspect of their bait elements) shown on the first page [**?**]. Previous work identified 44 clusters using a Deep Learning approach based on Convolutional Neural Networks (CNNs).

We develop our own CNN model to perform feature extraction, creating a feature space where visually-similar samples are mapped close to each other. The model takes a screenshot of the first page of each document and returns a 32-dimensional vector denoting its position in the new feature space. We create a training set starting from the one provided by [**?**]. We performed data cleaning when necessary, removing outliers and filtering or remapping elements to new groups based on their similarity. Finally, we augment it with more recent data from our data feeds, obtaining a total of 23,098 training samples divided into 47 groups. Next, we use a semi-hard triplet selection process and the triplet-loss function to train the model weights (see [**?**]). With this model, we extract a feature vector for each PDF and then apply DBSCAN [**?**] for clustering. To reduce manual intervention, we incorporate pre-labeled samples, or "anchors", into the pool of unseen documents. This way, we can automatically label the clusters based on the group of anchors they contain. If multiple anchors are associated with the same computed group, we re-cluster its samples using a smaller ϵ with DBSCAN until the conflict is resolved. Human intervention is only required when our *Clustering module* identifies a new cluster. **??** provides further details on the model and clustering procedure.

4. Characterizing Support Infrastructure

The goal of this section is twofold. Firstly, we examine the host and service composition, seeking similarities among hosts. Addressing this early on in the setup phase enables us to conduct specific analyses later on for these host types, which we study during the main phase. To tackle **RQ 1**, we investigate the network properties (Autonomous System, DNS lookup, URL) of the 1,350,201 URLs extracted from *Seed DS* (backlinks leading to to clickbait PDFs, see § **??**). Our analysis of these properties (§ **??**) reveals the presence of large groups of hosts with similar traits. Specifically, we observe three different types of hosting, covering 54 eTLD+1s. Next, in § **??** we run ad-hoc analyses on the websites serving 2,710,959 live clickbait PDFs during the Main phase of the study. We identify six plugins and two web frameworks facilitating file upload, and 12,927 origins hosting outdated software components, answering **RQ 2**.

4.1. Analysis of Network Properties

The goal of this section is to identify whether certain hosts within the supporting infrastructure share similar features, which we define in terms of network properties.

Autonomous System	# FQDNs	Autonomous System	# PDFs
WEEBLY, US	41,483	WEEBLY, US	241,851
AMAZON-02, US	9,222	*AMAZON-02, US*	142,200
WILDCARD-AS	5,351	CDN77 ^_^, GB	59,213
GOOGLE-2, US	4,301	*CLOUDFLARENET*	57,156
ZETTA-AS, BG	4,091	*GOOGLE-2, US*	46,264
AUTOMATTIC, US	1,556	UNIFIEDLAYER	37,504
CLOUDFLARENET	1,363	OVH, FR	34,974
OVH, FR	1,141	*GO-DADDY-CO*	31,080
IWEB-AS, CA	1,097	*ARUBA-ASN, IT*	25,731
UNIFIEDLAYER	1,086	FASTLY, US	25,703

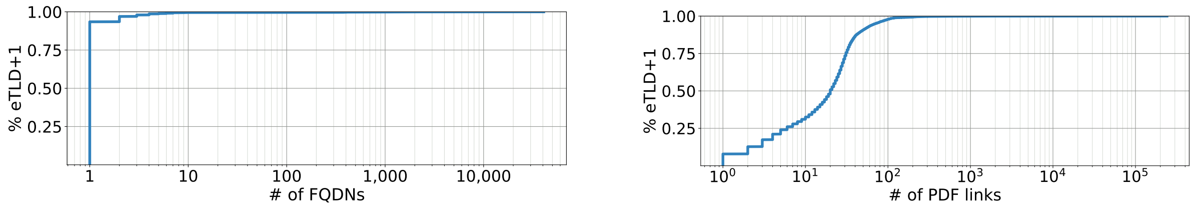
TABLE 2: Top ten Autonomous Systems sorted by number of FQDNs (on the left) and by the number of PDFs (on the right). The two lists report different AS names depending on their rank determined by the sorting criterion.

To find out if and which components make up the supporting infrastructure, we conduct an exploratory analysis of the *Seed DS* backlinks. Since attackers target large amounts of websites having the same security flaw (see, e.g., [**? ? ?**]) we analyze our data to find large groups of hosts sharing similar network properties. Our approach does not aim at identifying hosting provider *organizations* [**?**] but groups of similar Web hosts targeted by attackers.

4.1.1. Methodology. We focus on those indicators that can either be observed directly (e.g., domain name) or obtained via well-established channels (e.g., DNS queries). For example, given a URL `http://babemozigu.weebly.com/dir/file.pdf` we extract its FQDN (`babemozigu.weebly.com`) and its eTLD+1 (`weebly.com`), or "domain root". We obtain the IP address and the Autonomous System (AS) for each FQDN from the respective DNS and WHOIS records. For readability purposes, we aggregate different AS names belonging to the same company (e.g., `CLOUDFLARENET, US` and `CLOUDFLARESPECTRUM Cloudflare, Inc., US`, in italics) and report in **??** two distinct lists of the ten most affected ASes, independently sorted by number of unique FQDNs and by number of observed clickbait PDFs.

We noticed a significant difference in the order of ASes between the two lists. For instance, the ASes for `Weebly`, `Wilcard`, and `Zetta-AS` (first, third, and fifth ASes) were found to be the most frequently abused in terms of FQDN, but their overall rank differs considerably when sorting them by number of clickbait PDFs. **??** shows the distribution of FQDNs per domain root. The graph shows a sharp increase, indicating that the majority of domain roots (96%) have either no subdomain or just one subdomain. However, a small percentage ($< 0.01\%$) of domain roots have ten or more subdomains. To further analyze this, we set an empirical threshold of 100 FQDNs per domain root and manually investigate the resulting 20 domain roots. These eTLD+1s represent 97% of the domain roots with at least one subdomain. For example, `babemozigu.weebly.com`, `babewepuk.weebly.com`, and `babexunerasosib.weebly.com` are among them.

In the right column of **??**, we present different ASes based on the number of served clickbait PDFs. The dis-



(a)

(b)

Figure 3: (a) Distribution of FQDN per eTLD+1. (b) Distribution of `.pdf` links per eTLD+1. Data from the *Setup phase*.

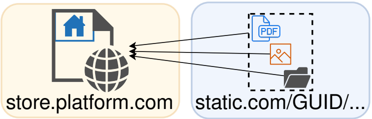


Figure 4: Example showing static resources residing on a different domain (PDFs in the *CDN* category).

tribution of PDF files across domain roots (**??**) shows that most eTLD+1s host a maximum of 100 clickbait PDFs, while only 2% of the domains serve more than that. We adopted a conservative approach to identify candidate domain roots by using the number of `.pdf` links as a criterion. As uploading a large amount of PDFs to a compromised website is easier than obtaining free subdomains, we set an empirical threshold of 5,000 PDF links per eTLD+1. We manually investigated the domain roots that exceeded this threshold in terms of PDF volume.

4.1.2. Results. This procedure identified 26 unique eTLD+1s. We confirm the existence of specific hosting services running on that domains by conducting a separate market research for services exhibiting similar characteristics. As a result, we identified three services running on these eTLD+1s, namely *Object storage*, *CDN* and *Website hosting*, which we explain below.

Object storage is a hosting service that manages unstructured data, such as PDFs, as individual units, or *objects*, stored in a single location [**? ?**]. The URLs of these objects include strings resembling unique identifiers, either as subdomains or in the URL path. Although these origins cannot be browsed, files can be retrieved using known URLs. We found one domain root belonging to this category, whose service includes a free tier accessible after thorough checks (e.g., providing a valid credit card number).

CDN origins exhibit a filesystem structure that resembles that of *Object storage* services, where PDFs (and other static resources) reside on a separate origin from where the main website operates, as depicted in **??**. Through our manual analysis, we were able to link all but two of them (`sqhk.co` and `f-static.net`) to a specific hosting service, such as E-commerce marketplaces or Shared hosting. During our market research on online hosting services, we discovered one entity using multiple eTLD+1s, as `s123-cdn-static-a.com` and `s123-cdn-static-b.com`. In our dataset, we identified five such instances and included them in this category.

In *Website hosting* services, multiple websites run on the same server. The services running on those domains observed in our data offer affordable options, including free subdomains or automated website building, an online service that enables users to create websites without coding skills by combining pre-designed modules. We verified that these services allow users to publish a website without requiring a credit card or a valid email address.

Hosting Type	# URLs
Object storage	166,356
CDN	595,385
Website hosting	853,514
Remaining URLs (*Undetermined hosting type*)	4,126,172
Education	204,679
Graphics Multimedia and Web Design	129,954
Computers Electronics and Technology	116,090
Web Hosting and Domain Names	99,165
Sports	96,612
Remaining categories	289,680
No category found by [**?**]	2,095,555

TABLE 3: Number of URLs to clickbait PDFs over hosting types or website categories. Data from the *Main phase*.

We perform two extra checks to ensure that no other hosting service with a lower volume of abuse went undetected. First, we investigate the remaining FQDNs via a third-party web analytics service [**?**], observing 23 additional domain roots classified as *Web Hosting and Domain Names*. We verify the correctness of this label before adding them to our *Website hosting* group. Additionally, we checked our URLs against a manually curated list of hosting services, finding two URL matches for `digitaloceanspaces.com` (DigitalOcean) and four URL matches for `storage.googleapis.com` (Google). However, we do not include them in our further analyses as the volume of URLs for these two providers is negligible with respect to that of other *Object storage* providers identified by our methodology (e.g. Amazon, 49,065). **??** reports the volume of clickbait PDFs per hosting type and category, while exhaustive details on the identified hosting services (as eTLD+1, volume of clickbait PDFs and FQDNs) are reported in **??** (Appendix). We observe that the coverage of our websites provided by [**?**] is limited (10% of all domain roots), which might be explained by the low rank of some websites or by their offline status. In the remaining, we refer to websites in none of the groups *Object storage*, *CDN*, or *Website hosting* as *Undetermined hosting type*.

4.1.3. Takeaways. In this section we addressed **RQ 1** by scrutinizing observable properties of URLs hosting clickbait PDFs. Our methodology identified a total of 54 domain roots (26 via analysis of network properties, five via manual analysis, and 23 via a third-party service [?]), which we have verified correspond to existing hosting types and services. For the scope of this paper, we organized them in three broad groups, *Object storage*, *CDN*, and *Website hosting*. Note that these names might not cover all the extensive services provided by major providers. For instance, *Website hosting* might involve Website Builder services, along with managed and unmanaged shared hosting.

4.2. Indicators of Compromise

In this section, we investigate factors which may have facilitated the upload of clickbait PDFs on the abused hosts, answering **RQ 2**. Our analyses are tailored to the characteristics of each hosting type, investigating Access Control Lists, presence and up-to-date status of software components related to website abuse, and plugins which we observed to lead to file upload. We observe the strong presence of outdated and vulnerable components on *Undetermined hosting type* websites, while *Website hosting* domains present a bare software stack which is rarely outdated. Finally, we summarize our main findings.

4.2.1. Experimental Setup. Different hosting types expose distinct properties, requiring the development of custom analyses modules for each type.

Firstly, when their URL is requested (e.g., via `HTTP GET`), *Object storage* hosts return "data units", and authorized users can upload new data via protocols specified by the service provider.

Next, we consider *CDN* providers and observe that domains in this category return an HTTP status code `403` when requesting the base path ("/") or any path segment preceding a PDF file. In fact, their filesystem structure cannot be inspected via simple `HTTP` requests, similarly to *Object storage* origins. Collecting data on the respective "storefront" of *CDN* origins is impossible because systematically linking *CDN* origins to their respective homepage domains is infeasible (see **??**). Consequently, we removed all domains belonging to this category from the analysis.

Conversely, websites belonging to the *Website hosting* or *Undetermined hosting type* categories can be inspected via regular crawling. We determine the presence of outdated or vulnerable components in two ways. First, we compile a list of server-side software components that previous works found to be connected to Internet abuse. These are: *(i)* type of web application, specifically CMSes and E-commerce software; *(ii)* their version (see, e.g., [? ?]); *(iii)* a list of plugins and themes, as the ones for WordPress, when applicable (see, e.g. [? ?]). *(iv)* the presence of Unrestricted File Upload vulnerabilities, as highlighted to be used in conjunction with SEO attacks [?]. Second, we performed a manual analysis of selected URLs, which led to the identification of eight additional components linked to file upload, for which we develop a custom scanner.

We follow best practices and disclosure guidelines in these analyses. Due to ethical concerns, we develop non-intrusive analyses looking for indicators of compromise (hereinafter IoCs), refraining from sending `POST` requests to verify vulnerabilities when this would trigger a state change on the target website.

SW Category	SW Name	# versions	# FQDNs
CMS	WordPress	188	4,041
CMS	Joomla	3	209
CMS	Drupal	3	112
Ecommerce	WooCommerce	150	1,310
Ecommerce	EasyDigitalDownloads	11	24
Ecommerce	Magento	1	4
Prog. language	PHP	280	8,206
Web servers	Apache	40	1,884
Web servers	Nginx	68	192
Web servers	IIS	7	438
WP plugins	Yoast SEO	193	1,463
WP plugins	WooCommerce	150	1,310
WP plugins	Revslider	115	623
WP themes	Astra	56	170
WP themes	Hello Elementor	8	71
WP themes	OceanWP	29	66

TABLE 4: Three most popular outdated software components per category.

4.2.2. Misconfigured S3 Buckets. The only *Object storage* service in our dataset corresponds to Amazon's Simple Storage Service. Thus, our analysis of *Object storage* websites is based on the collection of metadata on S3 buckets permissions. We develop our S3 scanner module relying on a popular library [?] on top of the AWS SDK. Similarly to [?], we proceed with the inspection of each bucket, collecting Access Control Lists (ACLs) and bucket contents when possible. For ethical reasons, we do not try to write any file to the buckets. We observed that a bucket may still exist even if one or more referenced PDFs are not online, thus, we feed the S3 scanner module all *Object storage* links, regardless of their online status. We probed 1,776 unique buckets in total, obtained from 159,403 links, where 243 were reachable at the time of scanning, while the remaining ones raised an error (e.g., `NoSuchBucket` or permission denied). We find that 67 of them have a readable Access Control List, where 21% of the buckets leave `Full Control` permissions, 28% of the buckets leave `Write` permissions, and 51% of the buckets allow to read a bucket's ACL (`READ_ACP` permission) to unauthenticated users.

4.2.3. Outdated Software Components. Next, we consider *Website hosting* and *Undetermined hosting type* websites. We proceed with a two-way approach: first, we collect data on the software components running at all *Website hosting* and *Undetermined hosting type* websites actively serving PDFs. When no data point has been collected for a domain, we randomly select $n \leq 20$ additional links from its home page and visit them, to increase the probability of triggering and detecting a vulnerable component.

We focus on software components of the following categories: Content Management Systems (CMSs), Ecommerce software, Hosting panels, Web servers, plugins and themes (as those of WordPress), and software components using the PHP programming language. We visited all FQDNs that served at least one clickbait PDF, i.e., 85,582 websites, and observed indicators relative to the above categories for 29% of them, identifying a total of

SW Component	# FQDNs		
	Path IoC	Scanned	% vulnerable
KCFinder	799	262	100
CKfinder	2,436	4,396	100
FCKEditor	232	4,933	0
CKEditor	88	4,840	91
Webform	482	-	-
Formcraft	621	-	-
SLiMS	1,018	396	73
E-Learning Madrasah	396	396	38

TABLE 5: Number of FQDNs running software facilitating file upload, with IoCs found in the URL path or via crawling.

299 software components. Next, we determine outdated software components by comparing their observed version on a target domain to their latest version at the time. We observe that most of the domains where this information is available are *Undetermined hosting type* domains (96% of the total observations), where more than half of these websites run outdated components. Conversely, only 26% of the software components observed on *Website hosting* domains are outdated. **??** reports the most popular outdated components per category.

As a last step, we inspected the network traces of our scanners to determine why no information was collected for a large amount of FQDNs. This inspection revealed that 90% of the websites that did not return any information are `weebly.com` subdomains, where the crawling was unsuccessful for `Timeout` errors as the IP was blocked. All the other domain roots were regularly visited by our scanner[1].

4.2.4. Vulnerable Software Components. We construct Common Platform Enumeration identifiers [**?**] using the retrieved software and version information (115 software components with version), and query the National Vulnerabiliy Database (NVD) [**?**] to obtain corresponding CVE information. We enrich this data with vulnerability information from the WPScan Wordpress Vulnerability Database [**?**].

Among these, we identified 26 software components whose version, at the time of our inspection, was vulnerable. We filtered out vulnerabilities less likely to be linked with clickbait PDFs (e.g., buffer overflow) and focused on "Unrestricted File Upload" vulnerabilities. In total, we observed ten vulnerabilities of this type affecting five software components among those we inspected. Among those domains with software and version information, 11,815 ran a component listed in either the NVD or the WP vulnerability database, and 225 of them had a UFU vulnerability, all of them belonging to the *Undetermined hosting type* group.

4.2.5. Software Facilitating File Upload. An exploratory manual analysis of *Website hosting* and *Undetermined hosting type* websites revealed the massive presence of specific vulnerable or misconfigured plugins which could be abused to upload files. In particular, we analyzed the URLs looking for recurring URL path elements on a large scale, with a volume sufficiently large for them to be considered as a deliberate target. Our intuition comes from the observation that large numbers of URLs can be grouped together by path segments, e.g., 119,662 URLs residing on 1,016 different domains share the path segment `wp-content/plugins/formcraft/`. A manual analysis of the most common URL path groups (we could confirm 19 unique URL path patterns inspecting 194 websites) led to the identification of eight CMS add-ons and two Web frameworks[2], all having associated CVEs or a public exploit in popular repositories (**??** reports details and vulnerabilities for each component, while **??** lists path segment indicators).

The presence of IoCs in the path of a URL may be an early indicator of the presence of vulnerable software, which however does not exclude the presence of the same vulnerable components on websites whose URL paths do not have such indicators. We determine that a website runs a vulnerable component by matching the source code and version string of the component against a regular expresssion[3]. We found specific `.txt`, `.js`, or `.html` files exposing plugin versions through exploit repositories, manual inspection of compromised websites, or by inspecting the source code of the eight components. We compiled a list 107 possible locations for these files, which our crawler visits. We ran this analysis for four plugins, i.e., CKFinder, KCFinder, CKEditor and FCKEditor (verifying the vulnerability for the other two plugins was not allowed, as it required sending `POST` requests.) Visiting all 107 potential IoC locations for the unseen *Website hosting* and *Undetermined hosting type* websites daily is an expensive operation, not to mention the traffic load imposed on the target websites. To reduce the dimension of the data in our daily analyses we *(i)* group domains by URL path (i.e., all path segments excluding the file name), as an identical server-side directory structure is a clear indicator of the presence of a shared server-side component, and *(ii)* visit ten randomly-sampled websites per path group. After two weeks, we inspect the results and remove all potential IoC locations that did not produce any match, lowering their number to 59.

We observed 9,800 websites mounting one or more of the four "CK" plugins, 55% of which were vulnerable. It is remarkable that these domains, all marked *Undetermined hosting type*, actively served a total of 190,258 PDFs. We adopted a similar approach to verify the presence of vulnerable components in the SLiMS and E-learning websites. **??** shows the amount of domains whose URL path contains an IoC on the left and the amount of domains scanned looking for a vulnerable software component on the right, where its vulnerability was confirmed by observing its software version.

4.2.6. Takeaways. The goal of this section was to identify features of the infrastructure hosting clickbait PDFs which may facilitate the upload of clickbait PDFs.

[1] We strived to reduce the load on target websites performing analyses only once per FQDN.

[2] The plugins CKEditor [**?**], CKFinder [**?**], FCKEditor [**?**], KCFinder [**?**], Formcraft [**?**], Webform [**?**], and the Web frameworks E-Learning Madrasah [**? ?**] (shipped with CKFinder) and Senayan Library Management System [**?**].

[3] For example, `FCKeditorAPI={ Version:'2.3.2', VersionBuild: '1082'}`

Firstly, upon collecting ACL information for 27% of all active S3 buckets, we observed that all of them allowed unauthenticated users to perform operations, e.g., via the `FullControl` or the `Write` permission. In the remaining cases, we found that most of the PDFs were offline or the buckets were non-existent by the time we visited them, which suggests the possibility of a prior cleanup action. Consistently with these observations, the buckets with observable IoCs counted 4,191 unique URLs leading to clickbait PDFs.

We crawled 31,724 *Website hosting* and *Undetermined hosting type* FQDNs successfully (e.g., no `Timeout` errors) and observed that 51% of them run outdated software components. Among them, the amount of domain suffering from Unrestricted File Upload is low (2%), hinting at the fact that this might not be the primary mean used by attackers to upload clickbait PDFs. In total, these domains served 1,075,835 clickbait PDFs.

Additionally, we confirmed that 16.4% of the 31,724 websites were running at least one component of the "CK" family, facilitating file upload, serving 190,258 clickbait PDFs. We underline that this is a lower bound of the possible websites running these components, as we reduced the amount of website scanned due to the large daily amount of scans otherwise necessary. The number of IoCs observed on URL path hints at a higher number of websites, i.e., 21.3%. Overall, our analyses observed indicators of compromise for 46% (1,251,059) of the URLs analyzed in § **??**.

5. Use of Support Infrastructure

Having identified the types of hosting most abused by cybercriminals and the solutions to upload clickbait PDFs on them, we proceed to measure the duration of this activity via the *PDF Status Check* module, answering **RQ 3**. These analyses are conducted on clickbait PDF links in the *Main DS*, having discarded those with an offline status. Next, we group these PDFs by visual similarity using our *Clustering Module* (see § **??**) and observe how these clusters distribute over the hosting types.

5.1. Duration of Abuse

We calculate the duration of the abuse as the mean uptime of each clickbait PDF hosted on a specific origin (with the granularity of a single day), as shown in **??**. Among the 54 domain roots identified as hosting services, we observed the live abuse of 38 of them (the PDFs hosted on the remaining 16 eTLD+1s were not online at the time we observed their URLs).

The average uptime for a single clickbait PDF is quite long, i.e., approximately five months. However, due to the continuous upload of new PDFs on the same hosts, the overall abuse of hosting services extends even further, averaging around nine months. It seems as if attackers persistently exploited these hosting services throughout our 13 months of observations, with 1,818 domain roots receiving new uploads for this entire period. The type of hosting providing the longest average PDF uptime is *Object storage*, where this value reaches six months.

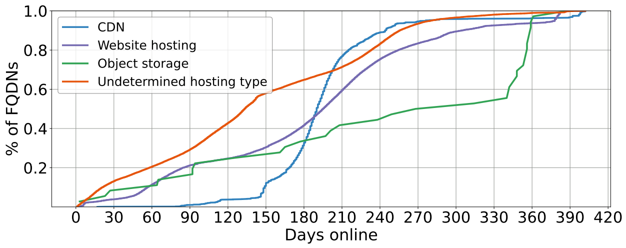


Figure 5: Distribution of clickbait PDF uptimes per hosting type, across our 13-month study.

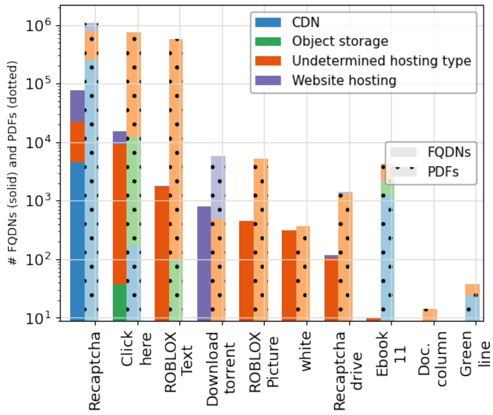


Figure 6: Stacked histogram showing clusters distribution across hosting types. Solid blocks represent the volume of FQDNs per cluster, while dotted blocks represent clickbait PDF volume.

5.2. Distribution of PDF Clusters on Hosts

In the *Main DS*, clickbait PDFs can be categorized, by visual bait similarity, into ten groups, seven of which align with those previously reported in [**?**], and four are newly identified. We observed fewer campaigns than [**?**], which could be attributed to attackers changing the visual baits used (since our data collection began 11 months after their experiments) or due to filtering out non-strictly-SEO campaigns. We gave the new clusters arbitrary names, i.e., *Click here*, *Doc. column*, *Green line* and *White*.

We use the group information to measure the distribution of live PDF clusters on different types of hosts, shown in **??**. When reading the graph by focusing on hosting types, we observe that all groups of clickbait PDFs make use of *Undetermined hosting type* spaces, although to different extents. Two groups (*Doc. column* and *ROBLOX Picture*) upload PDFs solely on this category of hosts. Two more (*ROBLOX Text* and *White*) rely almost uniquely on these domains, hosting there more than 98% of their samples. Conversely, PDFs belonging to the *Download Torrent* group are uploaded almost exclusively *Website hosting* hosts (91.7% of the samples belonging to this campaign). Finally, we observe that Amazon's S3 storage is the type of hosting targeted by the highest number of clusters, as we could observe six different ones[4].

Conversely, when focusing on the PDF visual clusters, we observe that they differ in how they use hosting types.

[4] *Click here*, *Ebook 11*, *Recaptcha*, *Recaptcha drive*, *ROBLOX Text*, *white*.

For example, the *Ebook 11* cluster tends to perform large uploads of PDF on few hosts, as there large imbalance between the number of FQDNs where PDFs are uploaded and the number of uploaded PDFs (approximately 150 PDFs per eTLD+1, see **??**). Differently, the *Click-here* and *Recaptcha* clusters distribute on average a smaller amount of PDFs per origin (approximately 2 per eTLD+1). A third example is that of *Download Torrent*, where there are large uploads of approximately 200 clickbait PDFs on two *Undetermined hosting type* domains alongside smaller batches of uploads on many *Website hosting* FQDNs.

5.3. Connection with IoCs

We also observed that 43% of clickbait PDFs belonging to the *reCAPTCHA* campaign and 52% of clickbait PDFs belonging to the *Roblox Text* campaign are hosted on websites running one of the targeted plugins (regardless of their observed version). These numbers represent a conservative estimate of the actual impact, as we chose to limit the number of IoCs tested to avoid excessive stress on the target websites.

6. Fighting Clickbait PDFs

We now evaluate solutions to counter the distribution of clickbait PDFs, tackling **RQ 4**. We first consider existing solutions, in the form of blocklists, evaluating the protection they offer to users. Our observations indicate that blocklists provide limited user protections, motivating the need to take action against the spread of clickbait PDFs. Our proposed solution involves the notification of affected parties, where we report our observations on the presence of clickbait PDFs and on the status of the components running on the websites hosting the PDFs.

6.1. Blocklists

In this section, we investigate whether common blocklists, as VirusTotal (VT) and Google SafeBrowsing (GSB), take action against clickbait PDFs by blocklisting their URL. This would offer a viable protection to users, which would then be protected when accidentally visiting the page of the PDF. We base our observations on 17 months of Google SafeBrowsing and VirusTotal daily lookups (i.e., since the beginning of this study).

We request scan results for 4 thousand clickbait PDF URLs daily to VirusTotal (approximately 50% of the daily amount) and receive a response in only 14% of the cases, where URLs are mostly flagged as malicious. This confirms the uncanny observation in [**?**] that URLs in clickbait PDFs are only partially scanned by VT and that this happens on the day the PDF is uploaded to the platform. When considering the type of hosting, we observe that VirusTotal flags domains belonging to all four of them, with *Website hosting* having the highest average rank (five AV engines) and *Object storage* having the lowest average rank (one AV engine).

Next, we observe that the number of URLs blocklisted by GSB is low, i.e., 0.4%. These URLs belong to 451 domains, with a mean ratio of URLs per domain of 41 (min 1, max 1377), which suggests that GSB is taking actions against clickbait PDFs and their hosts, blocklisting entire directories, but on a very small scale. Additionally, 99.7% of the blocklisted URLs belong to *Undetermined hosting type* URLs, suggesting that GSB does not take any action against clickbait PDFs hosted on well-known, reputable domains. When considering the overall lifetime of a clickbait PDF, as measured by the *PDF Status Check* module, we observe that a significant amount (29%) of the blocklisted PDFs is still online, which leads to think that blocklisting does not always correspond to a cleanup action.

6.2. Vulnerability Notification

Our next goal is to evaluate solutions beyond blocklisting to help reduce the spread of clickbait PDFs. One way to protect victims from the attack and, at the same time, to reduce the effectiveness of the SEO attack is taking down the PDFs by removing them from their location at the host. We thus undertake a large-scale notification of the threat posed by clickbait PDFs to the affected parties. Our primary goal is to observe the responsiveness of the hosting providers, measuring the amount of PDFs taken down as an effect of our reports.

6.2.1. Setup of the Study. We designed the notification procedure following best practices in this field [**? ? ? ? ? ?**].

Selection of Contacts. On Dec 1st, 2022 we select 799,930 `.pdf` links found online by our *URL Analysis* module on the previous day and divide their FQDNs equally in Treatment and Control group (8,843 and 8,842 respectively). Then, we look up their IP addresses and proceed to collect WHOIS records, obtaining 32,302 email contacts for 12,043 IPs. If necessary, we prioritize contacts from the same record, selecting `abuse@` contacts when present, `hostmaster@` contacts otherwise (following RFC 2142). If none of them are available, we choose one randomly. We obtained no WHOIS record for 153 domains, thus, we generate "synthetic" contacts by combining the aliases `abuse@`, `info@`, `security@`, `hostmaster@` with the domain name.

Content and Timeline of Notification. The notification e-mail briefly explains the threat posed by clickbait PDFs, then lists up to three clickbait PDF links among those hosted on up to three FQDNs belonging to the addressee. As a possible mitigation, we suggest the removal of the reported files and recommend a revision of the software components running at those domains. A CSV attachment reports all clickbait PDFs links for all the domains belonging to the addressee. Finally, recipients are given the possibility to opt out of the study or reach back for any feedback. The full text of our notification message is reported in **??**.

Finally, we set a time window of 30 days, from Dec 1st to Dec 31st, 2022. We notified domains in the Treatment group once every ten days for a total of three times and notified the domains in the Control group at the end of the study. The choice for a ten-day time interval is motivated by the observations reported in [**?**] where, in spite of the 14-day interval between each reminder, the number of fixes does not increase after ten days.

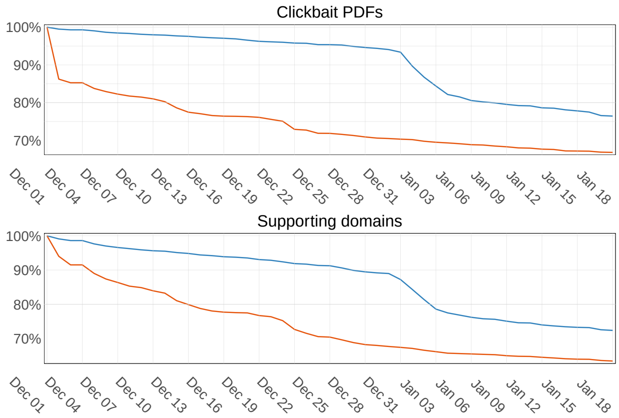


Figure 7: Takedown of clickbait PDFs and domains over time. The Treatment group is depicted in red and the Control group in blue.

Ethics. We did not seek IRB involvement for this procedure, addressing ethics concerns as follows. Contact points (participants) were chosen depending on the presence of clickbait PDFs on their domains. Participants were informed of the study and given the option to opt out immediately if in the Treatment group, or at the end of the monitoring period otherwise (Control group). Although vulnerability notifications might represent an additional overhead for security operators at hosting providers, the benefit gained from clickbait PDF takedown and a security review of the software stack outweigh this cost. To reduce recipient overhead, we grouped domains per abuse contact. Finally, we did not collect any user data and sought to increase privacy of operators and providers by processing answers per anonymous ID rather than email address.

6.2.2. Process. A final amount of 1,545 contact emails was selected as recipient for the notification. The discrepancy between the number of contacts and FQDNs stems from them sharing the same eTLD+1 or a provider managing multiple FQDNs. Due to a technical problem, 19 domains were not included in the reports or not reported at all, resulting in 1,522 emails being sent successfully. These contacts were notified together with those in the Control group, but removed from the reports.

As part of the notification process, we excluded one contact, who asked to stop the analyses of the PDFs residing on their domain. Moreover, we adopted a “cooperation policy” whenever explicitly asked, e.g., we re-sent the attachment or provided clarification on the threat (124 replies), acknowledged false positives (9 PDFs, $< 0.01\%$), or submitted a copy of the report via a Web form (25 submissions). Moreover, we estimated a lower bound of 257 contacts we never reached by inspecting the headers of bounced emails. As these providers could not be reached in the first round, we removed them from the Control group and did not notify them again.

6.2.3. Effectiveness. **??** shows the effectiveness of our notification by comparing the number of online clickbait PDFs in the Treatment and in the Control group. The remediation rates are 29.567% for clickbait PDFs in the Treatment group and 6.055% for those in the Control group, where their difference is statistically significant with $\rho < .001$ (estimated by using a Generalized Linear Model [**? ?**]). The number of online PDFs decreases sharply on the first days, while a less steep decrease is visible for the domains (**??**). One explanation for that may be that a few affected parties hosting a large number of clickbait PDFs took action immediately, while a larger number of entities, hosting less clickbait PDFs each, took longer to react. The low-but-existent remediation rate for the Control group suggests the presence of some form of “natural decay”, where a small fraction of clickbait PDFs go offline for causes not related to our notification. Nonetheless, the significantly higher remediation rate in the Treatment group shows an increased number of cleanup actions with respect to this phenomenon.

We observed that no affected party could remediate with respect to all reported domains (nor all PDFs, if on a single domain), and that 17% of the affected parties only partially remediated the notified issue. In particular, *(i)* 104 entities (7%) only removed those PDFs listed in the email body, ignoring the attachment. *(ii)* 154 entities (10%) cleaned all or some of the reported PDFs. However, after the notification, we gained visibility into unseen (and unreported) clickbait PDFs hosted by them, which we observed stayed online. *(iii)* 319 entities (21%) performed a full cleanup and also removed any PDF observed after the notification.

We also monitored the presence of the reported PDFs on VirusTotal to observe if any of the affected parties submitted the PDFs as a result of our notification. Given the large amount of clickbait PDFs involved and the limited API quota available to us, we opted to randomly sample unique PDFs, in equal amounts from the Treatment and the Control group. We fetched 111,787 reports relative to notified clickbait PDFs. 57,042 of these returned a record, where a negligible amount of them (1%) was either first submitted or last seen after the start of the notification. The number of domains hosting these PDFs belong almost equally to our Treatment and Control group. Thus, it does not seem that submissions to VT were triggered by the notification.

6.2.4. Long-Term Effectiveness. We observed a moderate but positive response to the vulnerability notification in terms of PDFs that were cleaned up. Our notification message clarified the possible presence of additional, unreported PDFs and recommended security audits on the software running on the affected domains. We further investigated the long-time effects of our notification of the affected hosts, measuring how many of them still served clickbait PDFs, albeit unseen ones. The observation of online unseen clickbait PDFs on notified hosts can be attributed to either new uploads from attackers or a partial cleanup by the responsible entity. **??** shows the online status of PDFs served by the domains involved in the notification. Starting from Dec. 1st, 2022 and Dec. 30th, 2022, we notice an increase in PDFs going offline, which mostly remains constant after the notification period concludes. Simultaneously, we continue to register unseen PDFs on the same origins, and their volume keeps growing over time. This disheartening finding shows that attackers have, and will continue to have, a relatively stable pool of hosts to upload PDFs in support of their attack.

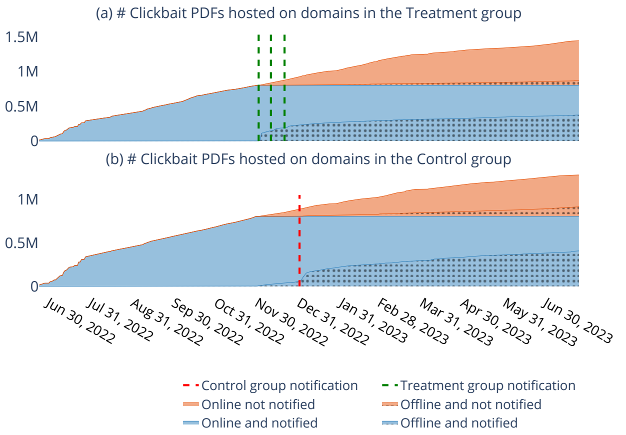


Figure 8: (a) Volume of PDFs in the Treatment group over time, online (solid color) and offline (dotted), versus new, unreported PDFs hosted by the same affected entities. (b) as for (a) but for PDFs in the Control group. (Control group volume is rescaled).

Our findings also suggest that disclosing the presence of clickbait PDFs is a moderately effective means of reducing the volume of online PDFs at a specific point in time. However, it proves ineffective in enhancing the overall security level of the affected hosts.

6.2.5. Feedback from Affected Parties. We observed two main types of reactions to the notification, i.e., appreciation and interest versus an uncooperative attitude.

Security Issues. A few affected parties confirmed our report and provided additional details or engaged in a conversation, allowing us to gain some invaluable insight on the issues they observed. Five of them confirmed that their clients were running the plugins we identified in § **??**, specifically plugins of the "CKFinder" family or Formcraft for Wordpress. Eight of them only generically replied that their client's CMS software was outdated (e.g., Joomla, Drupal) adding that they observed one or more PHP shells most likely used by the attackers to upload PDFs. One entity mentioned that their customer was running a custom web application. Finally, in three cases, the answers reported that the website seemed to be abandoned by the customer, who was also unreachable.

Not Phishing. Interestingly, one addressee answered all three notifications arguing that our report was unsubstantiated. They insisted that the reported PDFs did not pose any threat. Although we clarified the attack, they stated that they would not remove these legit files, as "*an interactive PDF with an attached hyperlink protected by recaptcha does not fall within the scope of phishing*", referring to PDFs reproducing the reCAPTCHA service to trigger a click.

7. Discussion

***SEO metric*.** Our *SEO metric* was designed with the goal of filtering out benign or not-clickbait PDFs to avoid processing personal data or poison our SEO-focused dataset. We perform a manual inspection to confirm that it only selects clickbait PDFs by inspecting up to 500 PDFs, fitting the SEO metric and randomly sampled from each cluster, for a total of 3,000 PDFs. The manual analysis confirmed the null number of false positives. Conversely, some clickbait PDFs with too few backlinks may be excluded. In the worst-case scenario, where all PDFs failing the *SEO metric* are clickbait PDFs, the false negatives would amount to 4.6% of 4.6 million links. We inspected 1,000 PDFs failing the *SEO metric*, randomly sampled from the *Seed DS*, and observed a much lower amount of false negatives due to the presence of benign or non-clickbait PDFs.

Development of *Grape*.The *PDF Status Check* module is a core component implementing the daily monitoring of online clickbait PDFs and enabling further analyses. We ensured the reliability of its results by repeating requests to endpoints leading to an error three times, or by using a VPN service. An interesting observation emerged where, in rare cases, an origin returned a different `HTTP` response for the same `.pdf` link. Specifically, we found that the `Content-Type` header differed between the `HEAD` request (not `application/pdf`) and the `GET` request (`application/pdf`). We examined a sample of 359 `.pdf` links marked as offline over the course of a week and did not observe any inconsistencies in the reported status. Moreover, we observed one origin cloaking the content of the `HTTP` response, i.e., serving clickbait PDFs only when visited by a browser instance with enabled JavaScript, and two origins protected by the CloudFlare Bot Management service. *PDF Status Check* does not intend to bypass bot protections, and interestingly, we observed that such mechanisms are notably scarce in prevalence.

Identification of Hosting Types. Our procedure for the identification of hosting types is based on observable metrics and indicators. All domain roots identified by our procedure correspond to an existing hosting service, confirming the validity of our methodology. We enriched this finding with the domain roots obtained from [**?**], which we verified belong to regional hosting providers. We cannot rule out the possibility that attackers might also abuse other types of services to a lesser extent. For instance, `documentcloud.org`, a document sharing platform, served 121 clickbait PDFs at one point. However, we did not come across any further instances of such activity.

Indicators of Compromise. Our findings show a grim picture of the landscape of software components running on the hosts part of the supporting infrastructure. Outdated and vulnerable components are especially present in *Undetermined hosting type* origins, whose software stack is likely not managed by the service provider. Ethical concerns on the traffic generated by our analyses on these origins limited the amount of scanning we performed to determine the component likely exploited by attackers to upload clickbait PDFs. Therefore, we believe the measurements we presented to be a lower bound of the amount of outdated or misconfigured software.

Vulnerability Notification. Our vulnerability notification procedure effectively reduced the number of clickbait PDFs supporting the SEO attack and provided valuable insights into the software components running on a few notified websites, corroborating our automated analyses.

One methodological choice in this procedure may have influenced its outcome. Specifically, we formed the Treatment and Control groups based on domains instead of contact points. This decision aimed to achieve granularity in measuring remediation, focusing on individual PDF files rather than affected organizations or entities. However, we acknowledge that this approach might have increased the likelihood of cleanup for other websites in the Control group falling under the same entity's responsibility. Moreover, our "cooperation policy", driven by a commitment to a safer Web, could have potentially influenced our results in a positive manner. We believe this impact to be limited, as we only engaged with 6% of the contact points.

External Threats to Validity. Our measurements might paint a less severe picture of the supporting infrastructure due to our partial visibility of the clickbait PDF threat. We mitigate this issue by collecting data from multiple sources: we build the *Main DS* starting from the *Seed DS* and observe that 93% of the total samples are not shared. We believe that a complete picture of this ecosystem might be visible only to entities whose crawling and processing resources are far above ours.

Looking Forward. **??** investigates the effectiveness of large-scale vulnerability notifications to address clickbait PDFs' abuse of hosting resources and protect users. While this approach proves effective in reducing online clickbait PDFs in the short term, there may be alternative methods to combat their distribution at various stages. For instance, making it more difficult for clickbait PDFs to rank high in search results could increase the attack cost and reduce the overall phenomenon. An implementation of this strategy could involve adding a module to a search engine crawler. The vast information available to search engines could serve as a crucial vantage point in preventing clickbait PDFs from achieving high rankings. Future research on clickbait PDFs could investigate which aspects of these documents are useful for detection.

8. Ethical Considerations

We designed the experiments for this paper keeping a series of ethical concerns in mind. The daily scanning of online PDFs and indicators of compromise may raise ethical concerns. We followed established guidelines [**?**], which included minimizing the frequency and load of experiments whenever possible (e.g., using `HEAD` requests instead of `GET`) and indicating the study's purpose, contact information, and opt-out option in the `User-Agent` header. Additionally, we conduct a manual analysis to focus on high-probability IoC endpoints, minimizing unnecessary scanning, and follow best practices in vulnerability disclosure, refraining from testing endpoints where this is not allowed (avoid verifying vulnerable endpoints when this requires sending state-changing `POST` requests). Finally, we reported all observed clickbait PDFs available with our large-scale vulnerability notification, started on Dec 1st, 2022. Our notification text explained about the threat posed by clickbait PDFs and included our contact points; we further gave participants the possibility to opt out of the study at any time. We plan to conduct another notification campaign reporting the PDFs that are still online at submission time.

9. Related Works

We now review previous works connected to our study.

Infrastructure Supporting Web Attacks. Previous studies have shown the use of abused infrastructure to support various attack campaigns, including malware delivery [**? ?**] and attack webpages [**? ?**]. Nonetheless, these studies focus on the attack itself rather than studying how attackers use the support infrastructure, or they focus on a pre-determined list of hosting providers. Conversely, we uniquely investigate the supporting infrastructure without constraints on hosting service or provider. Our study shares similarities with Li et al.'s research [**?**], both exploring malicious Web infrastructure, but differs in methodology, with Li et al. focusing on topological features of host interconnections.

Clickbait PDFs. Our study builds upon Stivala et al.'s work [**?**] which described the visual baits, structure, and distribution method of clickbait PDFs. However, our approaches differ significantly in methodology and goals, as we examine attackers' use of supporting infrastructure, constructing an ad-hoc dataset, extracting information on (sub)domains and hosting types, and identifying vulnerable endpoints.

Vulnerability Indicators. Previous works evaluated website security posture by detecting improper security headers and outdated software (e.g., [**?**]), WordPress plugins (e.g., [**?**]), or misconfigured S3 buckets (e.g. [**?**]). Another line of work shows that search engines represent an alternative to Internet-wide scanning, as they can find indicators of vulnerable web servers [**? ? ? ?**]. Part of our work touches the area of vulnerability scanning, as we verify the presence of exploitable vulnerabilities in hosts serving clickbait PDFs. Specifically, we focus on identifying known security vulnerabilities or misconfigurations allowing file upload, which attackers might have relied on for the upload of clickbait PDFs. However, we do not aim to develop a general-purpose vulnerability scanner and limit our assessment to well-defined indicators.

Abuse Monitoring in the Hosting Market. Finally, prior works examined abuse and cybercrime concentration from hosting providers' perspectives, particularly focusing on shared hosting and managed software components (e.g., [**? ?**]). They explored the correlation between an unsafe security posture at hosting providers and website compromise. Our work partially touches on services in the hosting market but does not target specific *organizations* or providers and does not aim to establish statistically significant metrics linking clickbait PDF abuse to specific hosting services.

10. Conclusion

This paper presented a 17-month study on the hosts supporting clickbait PDF attacks, counting 177,835 hosts and 4,648,939 links to clickbait PDF. We observe that the websites supporting clickbait PDF attacks belong to different types of hosting, such as *Object storage*, *CDN* and *Website hosting* observed in our dataset, and that their continued abuse lasts nine months on average. Additionally, we developed hosting-type-specific analyses and identified six plugins and two web frameworks facilitating file upload clickbait PDFs files, and a large amount of websites running outdated software. Finally, we responsibly disclosed our findings via a large-scale vulnerability notification, observing a statistically significant decrease in the amount of online PDFs. Nonetheless, we observed that most of the notified parties either suffered from re-uploads or performed partial cleanups, as their domains kept serving clickbait PDFs after the notification. While a few parties took action against this threat, we observe that their impact is limited compared to the total volume of online PDFs and their hosting websites.

Appendices

A. The *Grape* Pipeline

A.1. Clustering Module. Creating a clustering algorithm for the number and nature of samples presented a challenge, due to the necessity of handling shifting visual baits appearance and identifying new clusters without an available GPU. The pipeline operates in two steps: first, a CNN extracts a 32-dimensional feature vector from each sample, then, we use multiple iterations of DBSCAN to obtain document clusters. The embeddings are extracted daily, while the clustering procedure is manually triggered by a human operator.

The model. The model takes the screenshot of the first page of the PDF as a $128 \times 128 \times 3$ matrix and returns a 32-dimensional vector. It consists of five convolutional blocks (a sequence of Convolution, BatchNormalization, PreLu, and dropout functions), three downsampling operations (MaxPooling), and two final FC layers. Additional details about the model are shown in **??**. For training, we used a contrastive triplet loss with a margin of 0.2, implementing a semihard online triplet generation approach, as described in [**?**].

Clustering. We use DBSCAN to group the PDF embeddings based on their appearance. To reduce the need for human intervention, we include a list of 20 pre-labeled items per group in the set of samples to be clustered. This list aids in automatically associating the clusters created by DBSCAN with existing known groups of visually-similar clickbait PDFs. DBSCAN starts clustering samples with a default $\epsilon_0 = 0.25$. If a computed cluster contains anchor samples from different campaigns, we reprocess its elements with $\epsilon_{i+1} = \epsilon_i - 0.01$ until the conflict is resolved. The clustering procedure is initiated manually by a human operator, who regularly inspects newly discovered clusters to verify the quality of the results.

Validation. We manually inspected 3,840 samples by selecting at most 500 random elements for each cluster. In total, we found 105 misclassified samples, resulting in an error rate under 3%.

B. IoCs

B.1. Software Components Facilitating File Upload. This section presents the eight software components whose poor security status may have facilitated the upload of clickbait PDFs on a website.

FCKEditor, CKFinder, CKEditor, KCFinder. FCKEditor was a rich text editor first developed and

ID	eTLD+1	# FQDN	# URLs	Host. Type
•	amazonaws.com	9	49,065	*Object storage*
•	strikinglycdn.com	1	54,052	*CDN*
•	f-static.net	1	47,931	*CDN*
•	sqhk.co	1	15,484	*CDN*
⊗	squarespace.com	1	13,000	*CDN*
•	shopify.com	1	11,994	*CDN*
•	s123-cdn-static.com	1	10,200	*CDN*
•	filesusr.com	3,241	9,829	*CDN*
•	mozfiles.com	1,741	2,366	*CDN*
M	s123-cdn-static-d.com	1	531	*CDN*
M	s123-cdn.com	1	139	*CDN*
M	s123-cdn-static-a.com	1	138	*CDN*
M	s123-cdn-static-c.com	1	134	*CDN*
M	s123-cdn-static-b.com	1	124	*CDN*
•	weebly.com	40,803	241,092	*Website hosting*
•	epizy.com	4,242	5,722	*Website hosting*
•	pbworks.com	1,005	4,255	*Website hosting*
•	wordpress.com	1,547	4,039	*Website hosting*
•	rf.gd	3,071	3,765	*Website hosting*
•	iblogger.org	1,617	1,975	*Website hosting*
•	22web.org	1,506	1,829	*Website hosting*
⊗	myhome.cx	20	1,220	*Website hosting*
•	getenjoyment.net	459	1,101	*Website hosting*
•	mywebcommunity.org	419	1,059	*Website hosting*
•	myartsonline.com	423	1,041	*Website hosting*
•	mypressonline.com	432	1,040	*Website hosting*
•	onlinewebshop.net	388	968	*Website hosting*
•	mygamesonline.org	408	958	*Website hosting*
•	sportsontheweb.net	406	951	*Website hosting*
•	scienceontheweb.net	376	951	*Website hosting*
•	medianewsonline.com	391	950	*Website hosting*
•	atwebpages.com	390	940	*Website hosting*
⊗	linkpc.net	4	896	*Website hosting*
•	66ghz.com	171	260	*Website hosting*
⊗	esy.es	1	208	*Website hosting*
⊗	wpengine.com	3	197	*Website hosting*
⊗	webhostmurah.com	1	113	*Website hosting*
⊗	gridserver.com	2	64	*Website hosting*
⊗	ovh.net	2	45	*Website hosting*
⊗	yolasite.com	44	44	*Website hosting*
⊗	hekko24.pl	1	42	*Website hosting*
⊗	webbazaar.com	2	35	*Website hosting*
⊗	000webhostapp.com	1	34	*Website hosting*
⊗	pokladnicka.cz	1	21	*Website hosting*
⊗	altervista.org	1	21	*Website hosting*
⊗	jpn.ph	1	20	*Website hosting*
⊗	leszno.eu	1	19	*Website hosting*
⊗	cafe24.com	1	18	*Website hosting*
⊗	tenten.vn	1	18	*Website hosting*
⊗	hostsolutions.ro	1	15	*Website hosting*
⊗	belonnanotservice.ga	1	8	*Website hosting*
⊗	home.pl	1	1	*Website hosting*
⊗	webd.pl	1	1	*Website hosting*
⊗	micron21.com	1	1	*Website hosting*

TABLE 6: Second-level domains and providers. Identification method (ID): • by threshold, `M` by manual analysis, ⊗ via Web analytics service [**?**].

Layer Name	Size In	Size Out	# Kernel
CNN-1	$128 \times 128 \times 3$	$128 \times 128 \times 8$	$(3, 3)$
CNN-2	$128 \times 128 \times 8$	$128 \times 128 \times 16$	$(3, 3)$
CNN-3	$128 \times 128 \times 16$	$128 \times 128 \times 32$	$(3, 3)$
MAXPOOL-1	$128 \times 128 \times 32$	$32 \times 32 \times 32$	$(4, 4)$
CNN-4	$32 \times 32 \times 32$	$32 \times 32 \times 64$	$(3, 3)$
MAXPOOL-2	$32 \times 32 \times 64$	$8 \times 8 \times 64$	$(4, 4)$
CNN-5	$8 \times 8 \times 64$	$8 \times 8 \times 128$	$(3, 3)$
MAXPOOL-3	$8 \times 8 \times 128$	$2 \times 2 \times 128$	$(4, 4)$
FLATTEN	$2 \times 2 \times 128$	512	
FC	512	128	
FC	128	32	
L2	32	32	

TABLE 7: Details of the model architecture.

SW Category	SW Name	# versions	# FQDNs
CMS	WordPress	189	5912
CMS	Joomla	3	879
CMS	Drupal	3	347
Ecommerce	Cart Functionality	0	1828
Ecommerce	WooCommerce	159	1491
Ecommerce	EasyDigitalDownloads	11	24
Hosting panels	Plesk	0	1029
Prog. language	PHP	280	18279
Web servers	Apache	73	15065
Web servers	Nginx	68	5592
Web servers	LiteSpeed	0	2013
WP plugins	Contact Form 7	53	2105
WP plugins	Yoast SEO	196	1776
WP plugins	WooCommerce	159	1491
WP themes	Astra	57	203
WP themes	Hello Elementor	9	87
WP themes	OceanWP	31	83

TABLE 8: Three most popular software components per category.

released open-source by Frederico Caldeira Knabben in 2003 [**?**]. In January 2008, he released the first version of CKFinder [**?**], "the advanced file manager for FCKEditor" [**?**]. FCKEditor has been assigned eight CVEs, among which `CVE-2006-2529`, affecting all versions until 2.3 Beta, allows an attacker to upload files of any type. CKFinder has been assigned two CVEs, among which `CVE-2019-15862`, affecting all versions until 2.6.2.1, allows an attacker to upload files of any type. In 2009, the author renames FCKEditor to CKEditor, releasing for the first time CKEditor 3 [**?**] and founding CKSource Holding LTD. The development of FCKEditor was discontinued. Later, in 2015, right before the release of CKEditor 4.5, the plugin allegedly counted 15 million total downloads [**?**]. CKEditor has been assigned `CVE-2015-9349` for a Cross-Site Scripting (XSS) vulnerability affecting all versions before 4.5.3.1. A popular exploit repository has shared the code to open a reverse shell in websites running CKEditor 4.4.7 or earlier [**?**].

Finally, KCFinder was developed independently by Pavel Tzonkov [**?**] as a replacement to CKFinder, and to be compatible with FCKEditor and CKEditor. Its source code is still available [**?**], although archived in 2021. KCFinder has been assigned three CVEs, two of which are due to an XSS vulnerability and allow an attacker to inject and execute scripts. Affected are versions 3.20 and earlier, i.e., all versions. Multiple exploit repositories shared the code to exploit multiple vulnerabilities, e.g., Arbitrary File Upload in version 2.2 [**?**], Shell Upload in version 2.53 [**?**].

E-Learning Madrasah. This Web application was developed by the Indonesian Government as a response against the stop of all educational activities during the Covid-19 pandemic [**? ?**]. Educational institutions (e.g., high schools) were equipped with an online platform ("E-learning Madrasah") allowing all remote teaching activities. This platform comes with the vulnerable component CKFinder installed, whose exploit code is publicly available [**?**].

Senayan Library Management System (SLiMS). This is an open-source web framework for library management developed in Jakarta. Its popularity might be higher in Indonesia, as all websites mounting this framework have a `.id` country code. Moreover, the manual analysis showed that most of these websites were websites of educational institutions. SLiMS 7 and SLiMS 9 have been found vulnerable of multiple XSS, receiving two and three CVEs respectively, whose exploits are published in popular exploit databases [**? ?**].

FormCraft, Webform. FormCraft is a WordPress plugin offering form building functionalities [**?**]. Webform is a form builder plugin built for Drupal [**?**]. FormCraft versions below 1.2.6 and below 3.6 have been assigned two CVEs for two XSS vulnerabilities, and a popular exploit repository published the code targeting FormCraft version 2.0 leading to Shell Upload [**?**]. Conversely, Webform was found vulnerable to multiple vulnerabilities, including an XSS introduced by the inclusion of the vulnerable CKEditor library [**?**].

B.2. URL Path Indicators. Below is the list of indicators of compromise, where the URL path segments give out the presence of a possibly vulnerable component.

- SLiMS: keywords `__statics`, `gudangsoal` or `repository` in the URL path.
- CkFinder: URL path, param, query or fragment contain the keywords `ckfinder` or `ckimage` or `kcfinder` or `ckeditor` or `fckeditor` in the URL path.
- Formcraft: keyword `formcraft` in the URL path.
- WebForm: keyword `webform` in the URL path.
- SuperForms: keyword `super-forms` in the URL path.
- Formidable: keyword `formidable` in the URL path.

C. Notification email

I am a security researcher at `Institution Name` in `Country`. As part of an academic research project, we discovered that `N` of your domains (`domain1.com`, `domain2.com`, `domain3.com` among them) are used to host and distribute `M` clickbait PDF files. These files embed links leading visitors to malicious web pages delivering phishing attacks, malware, or online scams. Victims discover these clickbait PDFs with search engines such as Google and Bing, leveraging the reputation of your domains.

We do not know how exactly the attackers manage to upload these files in your domains and we believe that your domains may have a vulnerable or misconfigured component that enables unrestricted file uploads. Here is an example of three relative URLs to clickbait PDFs hosted by the above domains:

```
domain1:
/path/to/file/1.pdf
/path/to/file/2.pdf
/path/to/file/3.pdf
domain2.com:
/another/path/to/file/4.pdf
/another/path/to/file/5.pdf
/another/path/to/file/6.pdf
domain3.com:
/yet/another/path/7.pdf
/yet/another/path/8.pdf
```

/yet/another/path/9.pdf

We attach a CSV listing the clickbait PDFs relative paths per domain. Please note that the list we provided might not be exhaustive as attackers may have uploaded new files after this notification.

MITIGATIONS: As a first step, we encourage you to immediately remove these PDFs from your domains to hamper the effectiveness of the phishing campaign. However, we recommend a security review of your websites, looking for outdated, unpatched, vulnerable, or misconfigured software components to prevent attackers from uploading new files.

As part of our study, we will monitor the N domains to verify if they still serve such PDFs. You can opt out of this study by contacting us at author email. The details in this email should be sufficient for you to mitigate the problem, nonetheless, feel free to contact us at the same address should you have any question or feedback.

DISCLAIMER: This message is part of an academic research project. Researchers did not (and will not) attempt to reproduce the attack. We are not trying to sell any product or service, and we are not trying to obtain any bounty.