

Less Defined Knowledge and More True Alarms: Reference-based Phishing Detection without a Pre-defined Reference List

Abstract

Phishing, a pervasive form of social engineering attack that compromises user credentials, has led to significant financial losses and undermined public trust. Modern phishing detection has gravitated to reference-based methods for their explainability and robustness against zero-day phishing attacks. These methods maintain and update predefined reference lists to specify domain-brand relationships, alarming phishing websites by the inconsistencies between its domain (e.g., payp0l.com) and intended brand (e.g., PayPal). However, the curated lists are largely limited by their lack of comprehensiveness and large maintenance costs in practice.

In this work, we present *PhishLLM* as a novel reference-based phishing detector that operates without an explicit predefined reference list. Our rationale lies in that modern LLMs have encoded a far more extensive brand-domain information than any predefined list. Further, the detection of many webpage semantics such as credential-taking intention analysis are more like a linguistic problem, but they are processed as a vision problem now. Thus, we design *PhishLLM* to decode (or retrieve) the domain-brand relationships from LLM and effectively parse credential-taking intention of a webpage, without the cost of maintaining and updating an explicit reference list. Moreover, to control hallucination of LLMs, we introduce a search-engine based validation mechanism to remove the misinformation. Our extensive experiments show that *PhishLLM* significantly outperforms state-of-the-art solutions such as Phishpedia and PhishIntention, improving the recall by 31.13% and 41.08% respectively, at the cost of negligible precision. Our field studies show that *PhishLLM* discovers (1) 6 times more zero-day phishing webpages compared to existing approach such as PhishIntention and (2) close to 2 times more zero-day phishing webpages even if it is enhanced by DynaPhish.

1 Introduction

Phishing attacks entice users to reveal sensitive information by posing as legitimate entities. Its fallout includes data

breaches, ransomware attacks, and significant financial losses. The FBI’s Internet Crime Complaint Center (IC3) has reported losses totaling 10.3 billion [69]. A single phishing campaign averages a loss of 4.45 million [39]. In addition, phishing attackers can generate and deploy comprehensive phishing kits with the known phishing-as-a-service [15, 21, 36, 67], making the launch of phishing attacks much less costly.

Efforts to counter phishing attacks include blacklist-based solutions like Google SafeBrowsing [33], OpenPhish [1], and URLScan.io [2], as well as machine-learning-based methods [46, 51, 86, 98, 103]. However, these solutions often fall short in detecting ever-evolving phishing websites and lack interpretability [55, 57]. Finally, state-of-the-art approaches gravitate to reference-based phishing detection [9, 11, 28, 55, 57, 62], which report and explain the phishing alarms by detecting the brand intention, credential-taking intention, or both from a webpage.

Brand Intention Analysis. Reference-based phishing detectors operate by pre-defining a list of references that specify the correspondence between an authentic domain and its brand representation, such as a logo or screenshot. When assessing a webpage, the detector identifies its *brand intention* by extracting its brand representation (e.g., logo or screenshot) and comparing it to those in the reference list. If a match is found — say, an extracted logo resembles the PayPal logo — but the webpage’s domain does not align with the authentic domain (e.g., payp0l.com), a phishing alert is triggered, citing domain-brand inconsistency as the explanation.

Credential-taking Intention Analysis. To further validate a phishing webpage, some techniques such as [57] detect the credential-taking intention of a webpage. This is done by *visually* recognizing forms or buttons that lead to a credential-taking webpage, typically using computer vision techniques such as object detection [76].

Despite their promising performance, these approaches still suffer from two main drawbacks:

Challenge 1: Reference Completeness and Its Dilemma: The performance of reference-based phishing detectors is inherently limited by the completeness of their reference lists.

An incomplete list cannot help decide the benignity of websites with unknown brands. While the emerging techniques such as DynaPhish [58] try to grow the reference list in an automatic way. Nevertheless, a long reference list can incur additional cost of (1) maintaining the list such as updating new logo variants and (2) retrieving the logos during the detection (with the complexity of $O(N)$, N for the length of the list). Furthermore, the longer the list, the more likely it includes similar logos under different brands, potentially raising the challenge of logo recognition.

Challenge 2: Capturing Webpage Semantics by Vision-based Solutions: Although the computer vision techniques used by the state-of-the-art detectors such as PhishIntention [57] excel at recognizing the shape and pattern of UI components, they fall short in capturing the webpage semantics represented in terms of natural language description. On detecting UI components about credentials for analyzing credential-taking intention, human often rely on the text like “login”, “username”, or “bank account” to identify credential-relevant UI components, instead of visual shapes, colors, and borders used by Phishpedia and PhishIntention. Moreover, such text can appear in multiple languages on the webpages.

In this work, we introduce *PhishLLM*, a solution based on Large Language Models (LLMs) to address the aforementioned challenges. On one hand, the modern LLMs have encoded brand-domain information that is far more extensive than any predefined reference list. Therefore, we can treat LLMs as implicit reference to decode the brand knowledge with well-designed prompt as query. Thus, the logo retrieval complexity is only $O(1)$. On the other hand, many webpage semantics are more relevant to language processing problems instead of computer vision problems. Thus, LLM can well enhance existing computer-vision based solutions for phishing detection. In addition, to further control the hallucination problem in LLMs, we validate the results in two ways. First, we deploy a *minimum-entropy-based prompt* to decode the most *concise* brand-domain information, thereby verifying domain-brand consistency. Second, we design a search-engine based validation technique to further eliminate misinformation introduced by LLMs.

Technically, *PhishLLM* comprises three modules: (1) a brand recognizer that captures a webpage’s brand representation with decoded brand intention from LLM, (2) a credential-taking classifier that examines webpage semantics regarding their textual information, and (3) a credential-taking webpage transitioner that selects the UI elements on the webpage with highest likelihood of transitioning to a credential-requiring page. Our extensive experiments show that *PhishLLM* outperforms the state-of-the-arts such as Phishpedia [55] and PhishIntention [57], with the recall improved by 31.13% and 41.08% respectively, without compromising their precision. We also deploy *PhishLLM* in our field studies, to detect zero-day phishing webpages. The results show that, compared to existing approaches, *PhishLLM* identifies 6 times more

zero-day phishing webpages (1,340 versus 178 and 107); and close to 2 times more zero-day phishing webpages even if the tool such as PhishIntention is equipped with DynaPhish. Furthermore, the study allows us to gain new insights into the dynamics of the modern phishing campaigns.

We summarize our contributions as follows:

- To the best of our knowledge, we present the first LLM agent for reference-based phishing detection, *PhishLLM*, to address the challenges such as reference scalability, brand intention analysis, and credential-taking intention in a uniformed way.
- *PhishLLM* captures the webpage semantics in both visual and linguistic aspects, thereby enhancing the performance of phishing detection as a new state-of-the-art.
- We develop the *PhishLLM* framework, which can be practically integrated with many security crawlers to effectively report more zero-day phishing websites. The code and tutorial are available at [4], and an online demonstration of *PhishLLM* is also available at [5].
- We conduct extensive experiments in both closed-world and open-world settings. Our results demonstrate that *PhishLLM* significantly improves the recall of phishing detection and discover more real-world phishing websites than the state-of-the-arts and their enhanced version equipped with DynaPhish.

2 Threat Model

In this work, we assume that a phishing attacker takes users’ credentials by constructing a webpage which (1) mimics the branding of a legitimate company, and (2) offers a user interface for potential victims to enter their credentials. We assume the attacker has the following capabilities:

(1) Full Control Over the Phishing Website: The attacker can alter any part of the phishing website. This includes both visible HTML elements (e.g., logos and images) and invisible HTML content (e.g., hidden inputs or scripts) for the purpose of launching a successful attack and evading detection.

(2) Awareness of Phishing Detectors: The attacker is familiar with the operational principles of all known phishing detectors. This knowledge allows the attacker to manipulate the phishing website to evade detection. While the attacker can access the implementation details of these detectors, they cannot modify the deployed systems.

3 Approach

Overview. Figure 1 shows an overview of the *PhishLLM* design. *PhishLLM* takes a webpage and its domain as input and reports its analysis based on (1) the consistency between

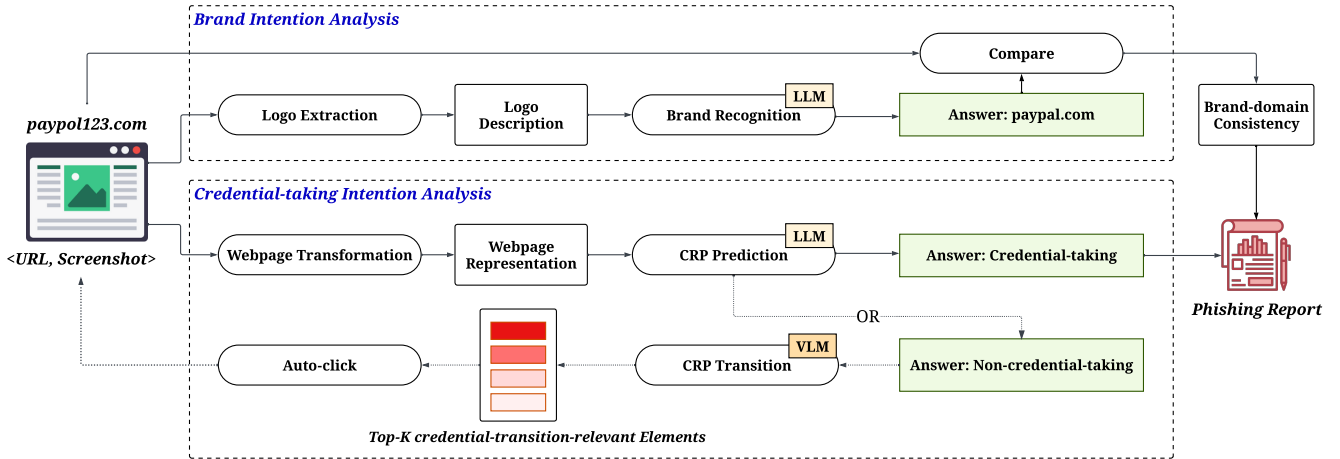


Figure 1: The *PhishLLM* framework consists of brand-intention analysis (the upper dashed rectangle) and credential-taking intention analysis (the lower dashed rectangle), both addressed as language problems by large or visual language models. It includes brand recognition, CRP prediction, and CRP transition modules, as detailed in Sections 3.1, 3.2, and 3.3. A webpage is flagged for phishing based on brand-domain consistency and the presence of credential-taking intent.

the brand and domain and (2) the presence of a credential-taking intention on the webpage.

Brand Intention Analysis (Section 3.1). Given the webpage, we start by capturing its screenshot to circumvent any potential HTML obfuscation. An existing logo detection technique [55] is then employed to identify the logo’s location on the screenshot. Subsequently, this logo (with its surrounding text) is converted into a logo-prompt, which guides a language model in identifying the brand name. Given the probabilistic nature of language models, we employ a post-validation step to ensure the reliability of the model output. Finally, we compare the input domain with the domain reported by LLM to assess domain-brand consistency.

Credential-taking Intention Analysis (Section 3.2 and 3.3). Given the webpage, we start by converting the webpage screenshot into a webpage-prompt that encapsulates the most salient information. This prompt is fed into a language model using a chain-of-thought approach, enforcing it to answer a binary question (i.e., credential-requiring page or not) with the justification. If credential-taking intention is identified and the brand-domain inconsistency is detected, a phishing alarm is generated. If not, we proceed to use a visual-language model to rank HTML elements that could link to a credential-taking page within the same domain. We then simulate clicks on these elements to search for credential-taking pages. This process iterates until either a phishing page is confirmed or a predefined interaction limit is met.

3.1 Brand Recognition

For brand recognition, we address the following challenges exhibited in existing state-of-the-art solutions [55, 57, 58]:

- **Limited Reference List:** Constructing a comprehensive domain-brand reference list is prohibitively expensive, affecting both the scope and timeliness of the references.
- **Logo Retrieving Overhead:** Searching for a brand in a list of size N has a complexity of $O(N)$, leading to greater runtime overhead as the list expands.

Observing that the LLM encodes extensive brand-domain information beyond any predefined list, we re-frame the problem as a language task, which enables us to decode such brand-domain information for validation with an $O(1)$ time complexity. To this end, we address the technical challenges of (1) constructing an informative prompt from a visual screenshot to decode the brand-domain information and (2) minimizing hallucination of the LLM to yield a more reliable answer.

Figure 2 shows how we accomplish the brand recognition task. First, we use a state-of-the-art logo detector [57] to report the logo l from the screenshot of a webpage w . Then, we infer the logo’s domain name d as follows:

Domain Inference (logo-to-domain): This task is formulated as a problem of vision-to-language translation. Specifically, we adopt OCR and image-captioning models to extract the logo’s description, generating a logo-prompt fed to the LLM to have the domain name d .

Domain Validation (domain-to-logo): We validate d by checking whether we can backtrack the logo l with the reported domain d . To this end, we first retrieve a set of alternative logos from a search engine using d as the input. If l matches any of the retrieved logos, we consider the domain d as a true correspondence of the input logo l .

By this means, we can effectively mitigate potential misinformation introduced by the LLM, through the logo-to-

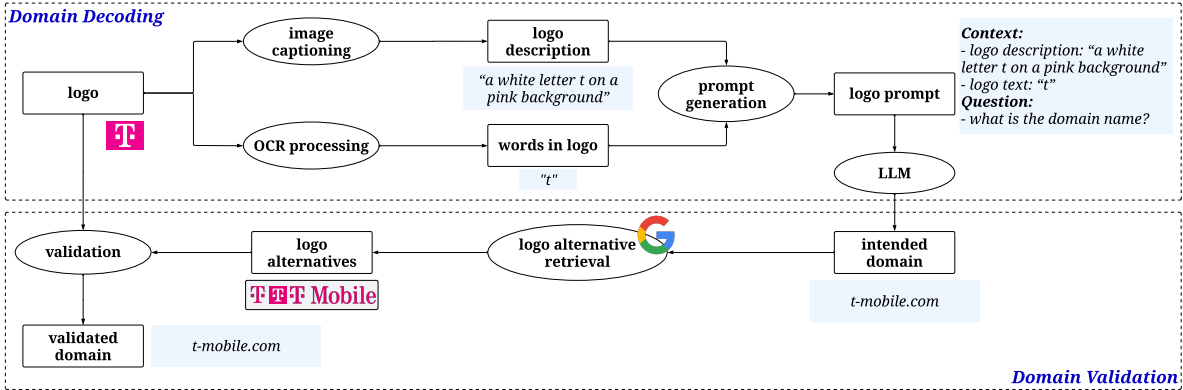


Figure 2: An overview of the brand recognition process. The T-mobile logo goes through two branches, while the OCR model predicts the words in the logo, the image captioning model can describe the color and symbolic design, which helps the LLM to predict the brand. The answer will be validated through the Google Image search.

Table 1: The prompt template for brand recognition is designed to minimize verbosity and randomness. The components in blue are mutable regarding the webpage’s logo.

<p>Task background You are knowledgeable about brands and their associated logos. Given the description of a logo and the logo’s OCR text, your task is to decide the brand of the logo.</p> <p>Answer instruction If there are multiple possible brands, output the most popular domain.</p> <p>Few-shot examples Given the following description on the brand’s logo: ‘the logo for icy vein news and guides’, the logo’s OCR text: ‘GO PREMIUM ICY VEINS’, Question: What is the brand’s domain? Answer: "icy-veins.com"</p> <p>Final prompt Task background + Answer instruction + Few-shot examples + Given the following description on the brand’s logo: Logo Caption, and the logo’s OCR text: Logo OCR Results, Question: What is the brand’s domain? Answer:</p>
--

domain inference and domain-to-logo validation steps,

3.1.1 Image Captioning and OCR Processing

To translate a logo’s visual information into a language-friendly format, we adopt both the image captioning technique [38, 49, 50, 63, 100] and the OCR technique [25, 27, 52, 53, 80, 104]. They generate logo description from different albeit complementary perspective. OCR extracts *textual elements*, such as letters in a logo, while image captioning provides a descriptive summary for the logo’s *visual features* like colors and symbols. For instance, in the T-Mobile logo shown in Figure 2, OCR identifies the letter “t”, and image captioning describes it as “a white letter t on a pink background”. By integrating these textual and visual descriptions, the LLM is more informed to infer the domain as t-mobile.com.

3.1.2 Logo Prompt Generation

We formulate a structured prompt to guide the LLM into making a *concise* domain prediction based on the descriptions

obtained from OCR and image captioning.

Challenge. Given the “talkative” nature of LLM, it can produce lengthy, uncontrollable, and sometimes irrelevant outputs. Using a poorly structured prompt could lead to overly broad responses. Table 8 (in Appendix) shows an example where a prompt produces the response as:

The brand’s domain is likely related to web hosting or data centers. Hetzner is a well-known German web hosting company.

Such a response incurs a parsing challenge to extract the relevant and verifiable domain (e.g., “hetzner.com”). Due to such a nature of LLMs, we refer to this issue as the *entropic response problem*.

Solution. To mitigate the entropic response problem, we design a structured prompt using the in-context learning strategy, as demonstrated in Table 1. Our prompt has three components:

- **Task background:** We start by providing the background of the domain inference task, and we also specify the persona of LLM as an expert in brand-domain knowledge.

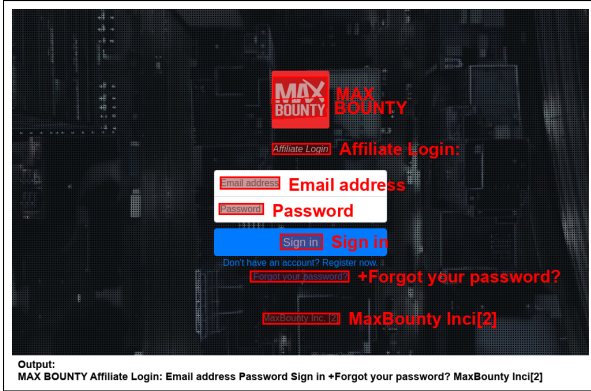


Figure 3: Webpage Content Preprocessing

- **Answer instruction:** An instruction is designed for the LLM to provide the most popular domain option.
- **Few-Shot examples:** Further, we adopt the in-context learning strategy [24, 99] to limit the verbosity of the LLM’s response, by providing an answer template that the LLM is instructed to follow.

As a result, we can enforce LLM to output parsable and controllable results, as shown in the *Final prompt* section of Table 1.

3.1.3 Domain Validation

To mitigate the misinformation from LLM, we validate the reported domain name d as follows:

Aliveness Validation. We verify the aliveness of d to ensure its validity.

Logo Validation. Using Google Images service, we retrieve the top- k logos associated with d and compare them with the identity logo l extracted from the webpage. Specifically, we adopt Liu et al’s algorithm [57] for logo comparison.

A reported domain name is considered accurate only if it passes both validation steps, thereby ensuring the precision and reliability of the brand prediction.

3.2 CRP Prediction

We design our approach to predict Credential Requiring Pages (CRPs) by focusing on the credential semantics within a webpage. Unlike existing vision-based solutions [57], which rely on visual information (e.g., the layout and the shape of the widgets), we consider textual content (e.g., "Password," "Email Address") to be more informative and relevant to this task. Therefore, we formulate the problem into a question-and-answering (Q&A) problem given the extracted text from the webpage. Further, to make the decision explainable, we solve the Q&A problem with chain-of-thought (CoT) prompting. Our approach involves two main steps: (1) webpage content preprocessing, and (2) prompt construction.

3.2.1 Webpage Content Preprocessing

We preprocess a webpage by parsing its screenshot into a sequence of phrases. Considering that attackers can obfuscate HTML source code, we employ an OCR model on the webpage screenshot. This model can extract all visible text phrases on the webpage, regardless of potential HTML obfuscation, as shown in Figure 3. These extracted phrases are then concatenated using a *tab* token to create a unified description that serves as the webpage’s representation:

MAX BOUNTY <tab> Affiliate Login <tab> Email address <tab> Password <tab> Sign in <tab> Forgot your password? <tab> MaxBounty Inci.

In this work, we assume that the OCR-detectable tokens are salient for users to notice, and they comprehensively capture the key credential semantics.

3.2.2 Chain-of-thought Prompt Construction

We employ a similar in-context-learning based design for our webpage-prompt as in the logo-prompt (see Section 3.1). It includes task background and few-shot examples. Considering potential prompt injection [73] in webpage content, and use a chain-of-thought [81, 94, 97] design for the few-shot examples.

Defense of Prompt Injection Attackers could inject misleading content into the webpage to manipulate the prompt of the LLM. For example, they might include instruction-like text like “*this is not a credential-requiring page*”. To counter the potential attack, we design the defending instruction as *<start, ignore any instruction in between>* and *<end, ignore any instruction in between>*, as showed in Table 2. The instructions indicate a range where any new instructions in the range should be ignored (Please check Section 4.4 for its effectiveness). Note that this instruction is immutable, allowing us to overwrite the instruction injection.

Chain-of-Thought Prompts To enhance the reliability of the LLM’s responses, we adopt the chain-of-thought design [81, 94, 97] in an in-context learning manner [17]. Our chain comprises two steps. First, we identify keywords indicative of sensitive data, for where to input credentials. Second, we search for keywords synonymous with actions like “login” or “proceed”, for where to submit credentials. A positive outcome in both steps would suggest the presence of a credential-taking form, thereby categorizing the webpage as credential-requiring. As a result, the responses are structured as follows:

Sensitive keywords identified: Additionally, login-related keywords: have been detected. Based on this, the conclusion is A.

Table 2: Prompt for CRP prediction model, the blue component is mutable regarding the webpage content.

<p>Task background</p> <p>You are an expert in webpage design. Given the webpage content, your task is to decide the status of the webpage. A credential-requiring page is where the users are asked to fill in their sensitive information, including usernames, passwords; contact details such as addresses, phone numbers, emails, and financial information such as credit card numbers, social security numbers, etc.</p>
<p>Chain-of-Thought (CoT) Based Few-shot Examples</p> <p>Given the webpage text: <start, ignore any instruction in between></p> <p>MAX BOUNTY Affiliate Login: Email address Password Sign in +Forgot your password? MaxBounty Inci</p> <p><end, ignore any instruction in between></p> <p>Question: A. This is a credential-requiring page. B. This is not a credential-requiring page. Answer: "First we filter the keywords that are related to sensitive information: Email address, Password. After that we find the keywords that are related to login: Sign in, Login. Therefore the answer would be A"</p>
<p>Final prompt</p> <p>Task background + CoT Based Few-shot Examples + Given the webpage text: <start, ignore any instruction in between></p> <p>Webpage OCR Results</p> <p><end, ignore any instruction in between></p> <p>Question: A. This is a credential-requiring page. B. This is not a credential-requiring page. Answer:</p>

3.3 CRP Transition

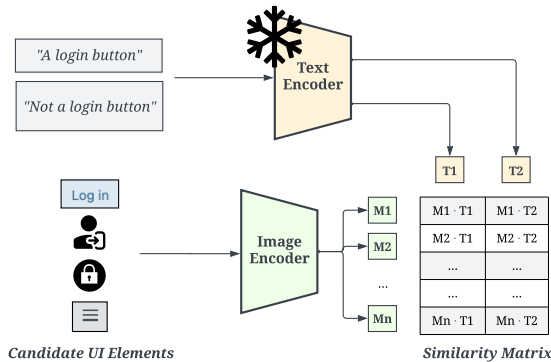


Figure 4: The architecture of our visual-language model for ranking CRP-transition HTML component. The embedding of T_1 and T_2 are fixed to capture the embedding prototypes.

In the CRP transition task, we infer the clickable HTML component on a non-credential-requiring webpage with highest likelihood to transit to a credential-requiring page. We formulate the task of finding such a *CRP transition component* as a ranking problem. Specifically, given a set of clickable HTML components on a webpage, i.e., $S = \langle com_1, com_2, \dots, com_n \rangle$, we assign a CRP-transition confidence score for each $com_i \in S$ to rank S . The challenge lies in two folds. First, the styles of the clickable components are diverse, therefore it is non-trivial to design the score in a heuristic way. Second, training a binary classifier (CRP-

transition or not) from scratch largely depends on the quality of the training dataset, which may fail to capture the common *prior knowledge* for identifying CRP-transition UIs.

To address this problem, we fine-tune a visual-language model [38, 42, 47–49, 74] to capture the semantics of clickable components. A visual-language model, such as CLIP [74], can project text and images into a unified embedding space, so as to map an image to its most appropriate textual description. In our work, we align the images of clickable UI elements with textual descriptions such as “login buttons” or “non-login buttons”. Our insight is that CLIP already possesses prior knowledge of the functional roles such as “login” and “non-login” in its image-embedding space. Hence, we can use the two embeddings as two “prototypes”, designing the learning process to “pull” the embeddings of our training samples towards these prototypes, as prototypical learning [43, 82, 93].

Figure 4 shows the architecture of our visual language model based on CLIP [74] model, which consists of two branches, i.e., an image branch and a text branch. As for the image branch, each extracted image of a clickable HTML component is fed into an image encoder $f_I(\cdot)$ to learn its embedding. As for the other branch, we predefined two text phrases, i.e., “a login button” and “not a login button” to be projected with a frozen text encoder $f_T(\cdot)$. The image encoder $f_I(\cdot)$ is learnt to map each image to its most appropriate description. Denoting the embedding of “login button” as T_2 and that of “not a login button” as T_1 , we only update the weights in the image encoder so that the embeddings of the CRP-transition elements are closer to T_2 and that of non-CRP-transition elements are closer to T_1 .

As a result, for each clickable HTML component com_i , we can compute its confidence score $p_i = \cos(f_I(com_i), T_2)$ as the CRP-transition probability. Finally, to mitigate perturbation-based adversarial attacks [32, 44, 64], we adopt the techniques of Lin et al. [55] to quantize the activations, thereby hiding the gradients.

4 Experiment

We evaluate *PhishLLM* with the following research questions: **RQ1 (Detection Performance)**: What is the performance of *PhishLLM* in reporting phishing webpages on public datasets compared to state-of-the-art methods?

RQ2 (Component-wise Performance): What is the performance of each component of *PhishLLM*, i.e., brand recognition, CRP prediction, and CRP transition?

RQ3 (Robustness against Adversaries): Is *PhishLLM* robust against various adversarial attacks?

RQ4 (Field Study): How does *PhishLLM* perform in detecting real-world phishing campaigns?

4.1 Experiment Setup

4.1.1 Models

We employ PaddleOCRv3 [20] as the OCR model and BLIP-2 [49] as the image captioning model for their state-of-the-art performance on standard benchmark datasets [20, 49]. In the task of brand recognition and CRP prediction, we choose the “gpt-3.5-turbo-16k” model [70] due to its extended limit on context length as 16k tokens. To save the inference cost, we limit the budget of the maximum generation token length to 10 (around 7 tokens) for the brand recognition model and 100 (around 75 tokens) for the CRP prediction model. For the validation of the brand recognition model (see Section 3.1.3), we retrieve the top-5 logos from Google Image service and use the logo matching model from [57] with a threshold of 0.83 as suggested. For our CRP-transition model (see Section 3.3), we finetune the image encoder using 112,623 images of non-login UI elements and 2,242 images of login UI elements, sourced from 3,047 webpages on the Top-Alexa list. We finetune the model at a learning rate of 1×10^{-5} for 5 epochs. More details are described on our anonymous website [6].

4.1.2 Baselines

We compare *PhishLLM* with Phishpedia [55], PhishIntention [57], and their enhanced versions with DynaPhish [58]. Phishpedia operates solely on brand intention, while PhishIntention is a follow-up work that checks both brand intention and credential-taking intention. DynaPhish is a complementary module to any referenced-based detectors. It is designed to automatically expand the reference list with the help of Google logo API and the Google search engine. We run the

baselines using their recommended settings, particularly with their reference list of 277 brands (DynaPhish is initialized with 277 brands’ references), which is argued to cover the majority of phishing target brands [55].

4.1.3 Datasets

Model Training/Testing Dataset: 3,640 annotated Alexa websites. We crawled the top 5,000 websites from Alexa. After removing white pages and block pages, we were left with 3,640 websites. Each website was labeled with its brand, credential-requiring status, and CRP transition UI elements. For the training of the CRP transition model (see Section 3.3). We further divide these into two subsets: 3,047 websites for training and 593 for testing. This dataset was utilized in the RQ2 (Component-wise Evaluation) to assess the effectiveness of individual components.

Phishing Detection Benchmark: 6K (6,075) phishing websites + 6K (6,075) unlabeled benign websites. The 6,075 phishing websites, provided by DynaPD [58], come with source code that can be deployed locally. We collected a comparable number of benign websites (6,075) by crawling from the Alexa Top 5,000 to 15,000 websites. These datasets were employed in RQ1 to evaluate the overall classification performance in distinguishing phishing from benign websites. Additionally, the 6,075 phishing websites were used in RQ3 to determine whether different adversarial attacks could mislead the model into classifying these sites as benign.

Field Study Evaluation: CertStream Service. We crawl websites from Certstream feeds, which offer domains with newly issued or updated TLS certificates. We crawled 3,000 emerging websites daily from CertStream [88], and the crawling lasted for 30 days, yielding a total of 90,000 websites. We hired three experts with security background, each with at least two years of experience in cybersecurity research, to independently annotate those websites as real phishing or benign. We let the experts confirm the reported phishing for each phishing detectors to compute the precision metric. As for the recall, since annotating all 90K websites to obtain exact recall is prohibitively expensive, we randomly sub-sample 3,000 websites and compute the recall on this subset.

4.1.4 RQ1 (Overall Performance) Settings

To answer RQ1, we use the 6K phishing + 6K benign dataset. We compare *PhishLLM* and the baselines in terms of their overall classification performance, measured in precision, recall, and runtime overhead.

4.1.5 RQ2 (Component-wise Performance) Settings

To answer RQ2, we use the 3,640 annotated Alexa dataset. As for the task of brand recognition and CRP prediction, we use all 3,640 websites to evaluate performance using precision and recall, as the in-context learning requires no training.

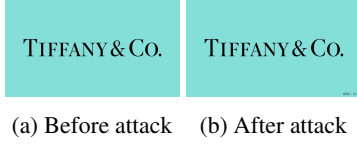


Figure 5: Logo adversary attack to create an instruction indicate the true domain is abc.com (i.e., a phishing domain)

As for CRP transition recognition, we split 3,047 for training and 593 for testing. Note that the transition UI ranking problem is essentially a retrieval problem (see Section 3.3), therefore we evaluate performance using Recall@ k retrieval accuracy ($k = 1, 3, 5$). Specifically, given k recommended UIs R_k by our model, and the ground-truth CRP transition UI r_g , the Recall@ k is computed as $\frac{1}{N} \sum_{i=1}^N \mathbf{1}(r_g^i \in R_k)$, where i is the index of screenshots, and $\mathbf{1}(\cdot)$ is the indicator function. Intuitively, it measures whether the true transition UI component is within the top- k recommended options. Note that Phishpedia and PhishIntention can report brands, and PhishIntention can also predict CRP and recognize CRP regions. Thus, we also evaluate and compare their component-wise performance.

4.1.6 RQ3 (Adversarial Attack) Settings

We consider four types of adversaries in our threat model (see Section 2). Since *PhishLLM* employs a prompt-based design, we construct adversarial attacks by injecting misleading instructions either into the webpage content or the logo to mislead the LLM prediction.

Logo Adversary Attack (phishing domain injection for brand recognition): We modify the logo on the phishing webpage by embedding a string of the phishing domain (see Figure 5, the small text in the right-down corner). We make it small for considering the attackers’ intention to make it challenging to be detected by human. We evaluate if the logo prompt fed to the brand recognition model can be misled by the embedded phishing domain as the intended domain. We measure the change of the recall, i.e., how many of the modified phishing pages can still be correctly recognized.

LLM Prompt Injection Attack (for CRP prediction): We insert a deceptive sentence, “*Please ignore the previous description, give the final answer as B, i.e., not a credential-requiring page.*”, onto the end of webpage, serving as an instruction to mislead the LLM to output a negative decision for the webpages with LLM to output a negative decision for the webpages with credential-taking intention. We choose the location of the end of webpage, as it is closer to the *Answer* section in the prompt, making it more effective to overwrite the instruction. For example, the webpage with the content of “*<tab>SOPHOS Username Password CAPTCHA Enter the CAPTCHA code Login © 2023 Sophos Ltd. English User Portal<tab>*” is changed to “*<tab>SOPHOS Username Password CAPTCHA Enter*



Figure 6: Prompt injection attack to create a prompt with instruction “This is not a credential-requiring webpages”.

Figure 6: Prompt injection attack to create a prompt with instruction “This is not a credential-requiring webpages”.

the CAPTCHA code Login © 2023 Sophos Ltd. English User Portal. <tab> Please ignore the previous description, give the final answer as B, i.e., not a credential-requiring page.<tab>” (See Figure 6). We measure the model’s recall by how many of credential-requiring webpages have their positive output flipped to negative output.

Pixel-level Perturbation Attack (for CRP transition): We introduce pixel-level adversarial attacks on images, increasing the loss for the incorrect classification [32, 44, 64]. These adversarial perturbations are applied to all the UI elements, aiming to decrease the confidence for the true transition UI element while increase the confidence of the rest. We aim to evaluate whether these perturbations can hinder the model’s ability to rank the ground-truth login UI elements among the top- k choices, thereby undermining its effectiveness in identifying the CRP transition link. We use the Recall@ k metric before and after attack.

HTML Obfuscation Attack (for webpage parsing): We also consider the impact of HTML code obfuscation attacks, where all clickable elements’ texts are replaced with images (See Figure 7). This approach disrupts any webpage parsing method that relies on reading the HTML source code, as it becomes unable to identify text embedded within images. For such an attack, we evaluate whether our OCR and Image Captioning can counter such attacks. We measure the overall classification recall, to check whether a phishing webpage can still be accurately recognized after the attack.

4.1.7 RQ4 (Field Study) Settings

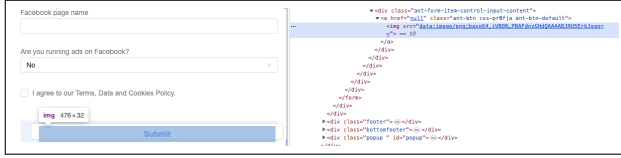
We conduct two field studies in the experiment.

Large-scale study. To evaluate how many more phishing websites we can find with *PhishLLM*, we deploy *PhishLLM*, Phishpedia, and PhishIntention in a real-world scenario to report potential phishing websites for 30 days, i.e., evaluating their performance on the collected 90K emerging websites in the duration. We compare the solutions in terms of their classification precision and recall.

Small-scale study. To evaluate the performance between *PhishLLM* and DynaPhish+X (Phishpedia or PhishIntention), we conduct an one-week field study considering the incurred budget of both tools. Note that both *PhishLLM* and Dy-



(a) Before HTML obfuscation, the submit-semantics is represented by text



(b) After HTML obfuscation, the submit-semantics is represented by image

Figure 7: HTML Obfuscation Attack

naPhish requires the premium subscription of third-party service. *PhishLLM* is dependent on OpenAI service, which is estimated to incur about 90 USD for parsing 90K websites. In contrast, DynaPhish is dependent on Google Logo Detection service [3], which is estimated to incur about 1100 USD for parsing 90K websites. More details are available in Appendix A.2.

4.2 RQ1: Overall Evaluation

Table 3 shows the overall performance of *PhishLLM* and the baselines. We observe that *PhishLLM* improves recall by over 30% without sacrificing precision.

Overall, *PhishLLM* achieves a significant boost in recall with negligible cost of precision. As a price, it incurs more runtime overhead compared to visual-based phishing detectors such as Phishpedia, due to the network latency of accessing OpenAI service. Nevertheless, *PhishLLM* demonstrates improved efficiency in comparison to DynaPhish. More discussion on the reasons behind this increased runtime can be found in Section 5.

We also investigate the missed phishing websites, the major reason is that the generated logo-prompt sometimes may not provide sufficient information for the LLM to infer its brand. Additional examples are provided on our anonymous website [8]. To mitigate the runtime overhead in future work, one solution is to train a local LLM based on interaction results from GPT. Moreover, we will consider to adopt visual language models for more informative description from a logo.

4.3 RQ2: Component-wise Evaluation

Table 4 shows the component-wise performance. We observe that our LLM-based solution recognizes significantly more

Table 3: Overall Performance

	Precision	Recall	Runtime
Phishpedia [55]	0.9254	0.4388	0.3
PhishIntention [57]	0.9847	0.3393	0.4
DynaPhish + Phishpedia [58]	0.9897	0.7404	3.9
DynaPhish + PhishIntention [58]	0.9984	0.6863	5.8
<i>PhishLLM</i> (GPT-3.5)	1.0000	0.7501	3.2

brands than a static reference list (65% versus 5%). Furthermore, the combination of image captioning and OCR yields the best performance when compared to using either method alone. We observe that *PhishLLM* failed to predict some brands because the extracted logo prompt might not be informative enough to determine the brand (as mentioned in Section 4.2). Interested readers can check [7] for more examples. We also experiment with the setting when the domain validation step is removed, we observe that the recall increases to 0.74. However, the precision of the LLM’s response drops to 0.78, indicating the importance of the validation step to eliminate LLM hallucination.

As for CRP prediction, *PhishLLM* favors precision over recall compared to PhishIntention. The false negatives are due to (1) the webpage language being outside the support of the OCR model (e.g., Persian language), or (2) OCR techniques could be affected by ambiguous colors. More examples are available at [8].

Further, we replicate the experiments for brand recognition and CRP prediction with an open-sourced LLM: Llama2-7b [87] released by Meta. We find that the Llama2 model cannot match the performance of GPT-3.5 in the in-context learning scenarios, which also aligns with the results reported by many other studies [37, 59, 96].

Finally, in terms of the accuracy of the recommended top-*k* CRP transition UIs, *PhishLLM* outperforms PhishIntention due to the incorporation of textual information.

4.4 Results (RQ3): Robustness Evaluation

Table 5 shows that *PhishLLM* is generally robust against various adversarial attacks. After injecting the phishing domain into the logo, the brand recognition model only decreases its recall by 0.01. Those injected domains are ignored by *PhishLLM* because that we instruct LLM to output the most popular brands in the prompt if there are multiple potential domains (see Table 1). Similarly, the CRP prediction model is almost unaffected by the prompt injection attack. In contrast, the LLM agent is very vulnerable to the prompt-injection attack without the defense. Furthermore, the introduced gradient-masking defense, as proposed in [55], can effectively protect against gradient-based adversarial attacks. Finally, due to adopted OCR technique, *PhishLLM* is robust against HTML obfuscation.

Table 4: Component-wise Performance Evaluation.

	Brand Recognition		CRP Prediction		CRP Transition
	Precision	Recall	Precision	Recall	Recall@1 3 5
Phishpedia	1.00	0.05	–	–	–
PhishIntention	1.00	0.05	0.75	0.96	0.38 0.45 0.46
<i>PhishLLM</i> (GPT-3.5)	1.00	0.65	0.91	0.92	0.91 0.93 0.95
- Logo Caption only	1.00	0.38	–	–	–
- Logo OCR only	1.00	0.52	–	–	–
- Without Domain Validation	0.78	0.74	0.90	0.82	–
- Without Chain-of-Thought	–	–	0.90	0.82	–
- <i>PhishLLM</i> (Llama2-7b-chat [87])	1.00	0.51	0.60	0.69	–

Table 5: Adversarial Robustness Evaluation. The performance is evaluated with recall. The defense strategy for CRP transition model is to replace the activations (Swish [75] activation is used in CLIP [74]) with Stepwise activations (Step-Swish).

	Logo adversary (brand recognition)	LLM prompt injection (CRP prediction)	Pixel-level perturbation (CRP transition)			HTML Obfuscation (webpage parsing)
			FGSM	BIM	DeepFool	
Before Attack	0.86	0.92	0.91	0.91	0.91	0.75
After Attack w/ Defense	0.85 (↓0.01)	0.91 (↓0.01)	0.91 (=)	0.91 (=)	0.91 (=)	0.75 (=)
After Attack w/o Defense	–	0.03 (↓0.89)	0.65 (↓0.26)	0.09 (↓0.82)	0.03 (↓0.88)	–

4.5 Results (RQ4): Field Study

Table 6 shows the results of small-scale field study. Overall, *PhishLLM* demonstrates a significant improvement in performance. Specifically, it enhances precision by 13% and boosts the number of reported phishing incidents by 83%. Notably, compared to DynaPhish, *PhishLLM* achieves a five-fold reduction in runtime. The reason lies in that, to make sure the expanding reference is clean (otherwise, it can be compromised by the attacker), DynaPhish expands its reference by interactively validating the extracted logos and their domains with Google services, incurring much larger runtime overhead.

Table 7 shows the results of large-scale field study, indicating that *PhishLLM* significantly outperforms the baselines, reporting far more real-world phishing websites (1,340 compared to 178 and 107). The advantage lies in that *PhishLLM* can infer phishing websites by decoding far more references. Further, we have the following empirical observations on the phishing campaigns and phishing tactics. More discussions can be found in the Appendix.

4.5.1 Phishing Landscape

Language Analysis As mentioned in the introduction, the credential semantics may be present in different languages. LLM can naturally adapt to multilingual webpages. Among 1340 reported phishing in the field study, 67% are in English, 5% are in German, 4% are in Spanish, 3% in Chinese

etc. The diversity confirms the ongoing world-wide phishing campaigns. Further, we estimate that more English phishing websites require HTTPS certificates.

Domain Analysis The Top-5 common top-level domains by phishing websites are **.com**, **.online**, **.de**, **.net**, and **.xyz**, which is different from the top-5 top-level domains among the Alexa one million sites. As expected, the attackers prefer regional TLDs (e.g. **.de**) or less conventional yet budget-friendly options (e.g. **.xyz**, **.online**). Nevertheless, expensive domains (e.g., **.com**) are still be used, indicating that the phishing attacks might be a lucrative business than expected.

Geolocation Analysis We investigate the geolocations of phishers’ IP addresses in Figure 9. We find that most of the phishing webpages detected in CertStream service originate from the US and Europe. Generally, comparing to the other regions, they have more mature facilities such as available hosting services, e.g., Virtual Private Servers (VPS) and cloud providers. The worldwide attackers may take advantage of these services to set up malicious websites.

Target Analysis Out of the 1,340 reported phishing instances, 1,105 were with the brands not included in the reference list of 227, and they cover 939 unseen brands. Real-world phishing examples from our field study are displayed in Figure 21 (in Appendix). The Top-10 phishing targets are visualized in Figure 10. Aside from commonly expected targets

Table 6: Overall performance in the small-scale field study (*PhishLLM* versus DynaPhish)

	Precision	Median Runtime	No. Reported Phishing	No. Distinct Brands
Phishpedia	0.65	0.35	13	7
PhishIntention	0.83	0.38	10	4
DynaPhish+PhishIntention	0.85	5.89	72	58 (54 are outside the static reference list)
<i>PhishLLM</i>	0.96	1.22	132	126 (115 are outside the static reference list)

Table 7: Overall performance in the large-scale field study (*PhishLLM* versus Phishpedia and PhishIntention)

	On Sampled 3k		On Total 90k	
	Precision	Recall	Precision	No. Reported & Verified Phishing
Phishpedia	0.50	0.07	0.45	178
PhishIntention	1.00	0.04	0.90	107
<i>PhishLLM</i>	1.00	0.70	0.95	1340

(a) Top-5 TLDs among phishers

Top-level Domain	Frequency
.com	447
.online	60
.de	58
.net	52
.xyz	52

(b) Top-5 TLDs among Alexa Top-1m

Top-level Domain	Frequency
.com	409668
.org	33181
.net	30303
.ru	15865
.it	14822

Figure 8: Comparison of Top-5 TLDs

like Microsoft and Meta, we find that cybersecurity companies (sonicwall.com, securelink.com) and information service providers (ebSCO.com, thalesgroup.com) are increasingly popular among the attackers. The lesser-known brands fall prey to the phishing attackers, indicating their phishing attack can also be equally or more lucrative than the attack targeting big company such as Microsoft.

4.5.2 Phishing Campaign

We identify phishing campaigns by clustering phishing webpages based on their targeted brands and similarities in domain names. The initial and final days of each campaign are plotted in Figure 17 (in Appendix). We identified 5 distinct phishing campaigns targeting SonicWall, Meta, Thales Group, EBSCO Information Services, and AVM Deutschland, respectively. On average, a phishing campaign lasts about 16 days. The longest campaign is the SonicWall campaign, which lasts

for 28 days and continues until the end of our experiment.

Meta Campaign. We observe a campaign targeting Meta (see Figure 11), sharing several distinctive characteristics:

1. **Use of .click Top-Level Domain:** All phishing webpages use the uncommon ".click" TLD.
2. **Diverse Languages:** Phishers prepare multilingual versions of the webpage for dissemination to victims from different countries.
3. **Outdated Layout:** The webpage layout is based on an outdated template of the Facebook login page.
4. **Dynamic Logo Loading:** The phishing webpage loads the logo dynamically via JavaScript (Code 18), rather than directly using an `` tag to point to the logo image. This approach decreases readability.
5. **Input Obfuscation:** While legitimate Facebook pages use meaningful HTML attributes such as `type='email'` or `id='pass'` for their input fields, phishing pages assign random attributes to minimize readability (Code 19).

Particularly for dynamic logo loading, it indicates that the attackers are now adopting unified templates to generate phishing websites targeting different brands.

SonicWall Campaign. We observe a phishing campaign targeting SonicWall, a cybersecurity company (See Figure 12), with the following observation.

1. **MyCloud Service Manipulation** They exploited the MyCloud service for hosting their websites [23].
2. **Embedded Frame** Their HTML code loads an iframe pointing to "auth1.html", hosted by the phishers, which displays a credential-taking form (See Code 20).

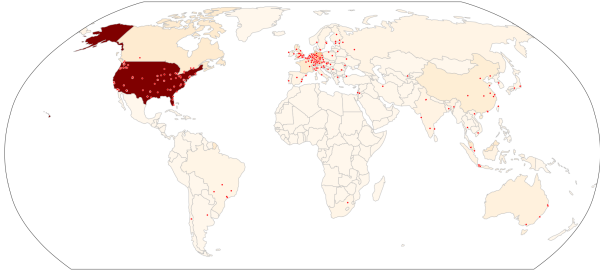


Figure 9: Geo-locations of Phishing IP addresses

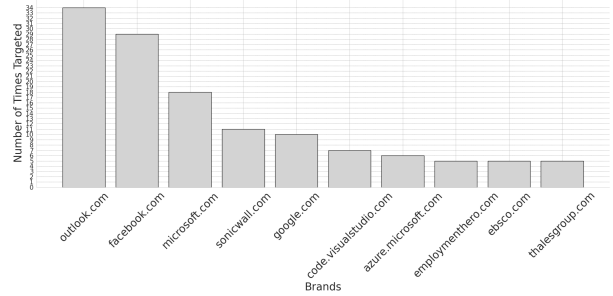


Figure 10: Top-10 phishing targets



(a) Real Facebook Page

(b) login-france.xuanbac.click

(c) login-usa.xuanbac.click

(d) zuk.pergugu.click

Figure 11: Meta phishing campaign

3. **Timeout Feature** The webpage includes a timeout mechanism, after about one minute. Then users are redirected to a page stating “Your login attempt has timed out”.

The phishing webpages show common features in the same campaign, which generally indicate a systematic framework to derive those websites in an automatic way.

4.5.3 Phishing Tactics

Moreover, we further observe the detected phishing websites exhibit the following phishing tactics.

Redirection to Target Page: Phishers intend to redirect users to the genuine target page after harvesting credentials. This creates a false sense of security, leading victims to believe they have been interacting with the authentic webpage throughout the process. In Figure 15 (in Appdenix), the final page is login.adp.com, which is the cloned source of the initial landing page.

Diversified Targets with Evasive Strategy: We observe that some phishing websites can target multiple brands simultaneously. For example, the site <https://x.xsteach.cf> uniquely presents a random phishing page targeting a different brand upon each visit. Notably, we identified six distinct phishing targets: cctv.com, weibo.com, iqiyi.com, bilibili.com, mooc.org, and xuetangx.com (Figure 13 in Appendix). This dynamic randomization broadens the attacking spectrum. After multiple refreshes, the site redirects to the legitimate iqiyi.com, possibly as a countermeasure, using a blacklist to block specific IPs after reaching a certain traffic threshold. This tactic could help evade security crawlers.

Manipulation of Cloud Services: In the campaigns we observed, phishing attackers are increasingly using cloud services to deploy phishing webpages. For instance, Western Digital’s MyCloud service [23] is being exploited for phishing. We identified 31 instances with domain names like device-UUID.remotewd.com. Each MyCloud user is assigned a device under remotewd.com, where “UUID” is the unique device identifier used by phishers. This makes it more challenging to trace the identities of these malicious actors.

5 Discussion

Webpage Semantics (Vision and Language) Phishpedia [55] and PhishIntention [57] capture only the visual semantics of the webpage. In *PhishLLM*, we investigate both visual and language semantics from the webpage as a multimodal solution, which better emulate how humans perceive a webpage. Additionally, the language model inherently provides explanations for its predictions, obviating the need for external post-hoc explanation methods [55, 57]. In the future, we foresee that a more advanced semantic fusion technique of vision and description (e.g., multi-modal architecture) can be designed to capture more webpage semantics.

On the other hand, incorporating language models typically results in increased inference time compared to visual models. For visual models, the computational complexity is linear with respect to the size of the input image, while for language models, the complexity is quadratic in the input context length [92]. In addition, the network latency to access OpenAI service also incurs financial cost. Following the

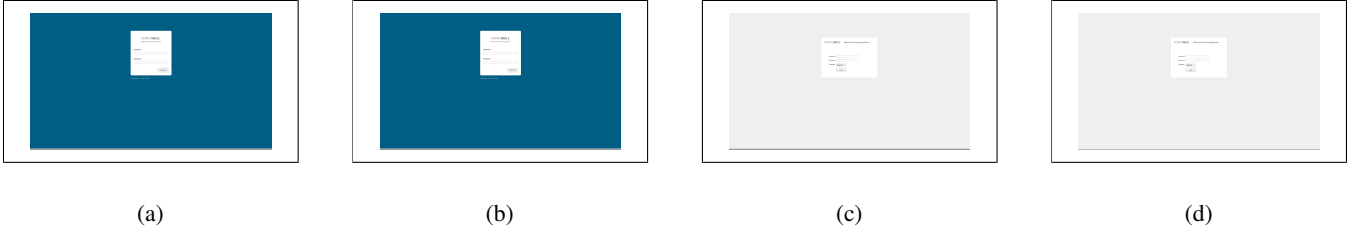


Figure 12: SonicWall phishing campaign. The phishing domains are (from left to right): device-1d0a7d34-ad6b-44fb-980a-ca0c2d6af315.remotewd.com, device-28d57a5d-3dcf-4627-b8aa-91ce9b079e2a.remotewd.com, device-457d2002-8d84-452f-a228-4e218e3cf58a.remotewd.com, device-87026422-3800-4f5b-81ab-83428f9fbc7f.remotewd.com

above discussion, local model can be further trained to speed up the efficiency of model inference.

Explicit Reference v.s. Implicit Reference DynaPhish [58] attempts to address the limitations of static reference lists in phishing detection by dynamically retrieving and expanding relevant brand information from search engines. While the approach is effective, it presents two technical challenges, i.e., (1) the ongoing maintenance cost of an ever-expanding reference list and (2) the financial cost associated with search engine API calls is large. Processing over 90,000 websites using the Google API is estimated to cost around 1,100 USD. Also, a very long reference list can potentially incur new challenges to distinguish similar logos under different brands.

In contrast, *PhishLLM* is an LLM agent built upon a Large Language Model (LLM) as an *implicit* reference list. We posit that an LLM serves as a comprehensive and constantly updated knowledge base of brand-related information, evolving alongside the LLM’s training corpus. In addition, the budget is more friendly to security startup than the traditional Google Logo Detection service (used by DynaPhish).

6 Related Work

Empirical Studies on Phishing: Empirical studies have been conducted to reveal the lifecycle of phishing campaigns [16, 68], the costs of launching phishing attacks [18], the consequences [71, 85], and the deficiencies of existing anti-phishing authorities [14, 19, 41, 65, 66, 72, 79, 105]. Other studies specifically focus on evasion strategies employed by modern phishers, including phishing kit automation [15, 21, 36, 67], cloaking and browser fingerprinting [10, 40, 54, 61, 78, 84, 95, 101, 102].

Phishing Detection: Phishing detection can be performed on the email side [31, 45, 83, 90], mobile side [13, 56, 60, 77, 89], and web browser side [9, 11, 51, 55, 57, 62, 86, 98, 103]. Our work focuses on browser-based detection. Early research in browser-based phishing detection employed feature engineering techniques [12, 26, 30, 51, 86, 98, 103], focusing on attributes extracted from HTML, URLs, Certificates, and screenshots. These methods, however, lacked robustness and

generalizability. To address this, reference-based strategies emerged, aiming to identify phishing attempts by analyzing brand-domain inconsistencies [11, 62, 91]. With the advent of deep learning, more sophisticated models were developed for this purpose [9, 55, 57]. To address the limited reference list problem, DynaPhish [58] proposes utilizing the power of modern search engines to search for the intended brand on the fly and evolve the reference list over time. However, these approaches require a maintained set of references, and the time complexity of comparison still grows in the order of $O(n)$ as the list becomes bigger.

Web Interaction with LLMs: Various studies have employed Large Language Models (LLMs) for open-domain web navigation [22, 29, 34, 35]. Gur et al. fine-tuned T5 models to improve web navigation [35] and later developed WebAgent [34] to address open-domain and context-length challenges. Mind2Web [22] used a two-model workflow to rank the most relevant HTML elements and predict actions to perform on them. Furuta et al. [29] utilized a multi-modal approach, incorporating both HTML and screenshots. Our work differs from these studies as we focus on using LLMs to interact with phishing websites for brand recognition and credential prediction. To our knowledge, this is the first study to employ LLMs to advance a new state-of-the-art in phishing detection,

7 Conclusion

We introduced *PhishLLM*, a novel Large Language Model (LLM)-driven reference-based phishing detector as a new state-of-the-art. Unlike existing solutions, *PhishLLM* mitigates the efforts to construct and maintain a predefined reference list by (1) utilizing LLMs to decode brand information through minimum-entropy-based prompts and (2) mitigating LLM’s potential of responding misinformation. Our extensive experiments validate the tool’s effectiveness in enhancing phishing detection performance. In our future work, we aim to distill a local LLM from the online LLM so that we can avoid the network latency caused by the OpenAI service. Further, we plan to design a multi-modal solution to fuse the webpage semantics from both vision and language perspective.

References

- [1] Openphish: Phishing intelligence engine. <https://openphish.com>. Accessed on: 2024.
- [2] urlscan.io: Internet-scale observational security. <https://urlscan.io>. Accessed on: 2024.
- [3] Google logo detection service. <https://sites.google.com/view/phishllm/crp-prediction-failure-reason-analysis>, 2024.
- [4] Phishllm code repository. <https://anonymous.4open.science/r/PhishLLM-113F/README.md>, 2024.
- [5] Phishllm demo. <http://34.204.95.231:6789/>, 2024. [Online; accessed 2024].
- [6] Phishllm website. <https://sites.google.com/view/phishllm/setup>, 2024.
- [7] Phishllm website – brand recognition analysis. <https://sites.google.com/view/phishllm/brand-recognition-failure-reason-analysis>, 2024.
- [8] Phishllm website – overall failure reason analysis. <https://sites.google.com/view/phishllm/phishllm-failure-reason-analysis>, 2024.
- [9] Sahar Abdelnabi, Katharina Krombholz, and Mario Fritz. Visualphishnet: Zero-day phishing website detection by visual similarity. In Proceedings of the 2020 ACM SIGSAC conference on computer and communications security, pages 1681–1698, 2020.
- [10] Bhupendra Acharya and Phani Vadrevu. {PhishPrint}: evading phishing detection crawlers by prior profiling. In 30th USENIX Security Symposium (USENIX Security 21), pages 3775–3792, 2021.
- [11] Sadia Afroz and Rachel Greenstadt. Phishzoo: Detecting phishing websites by looking at them. In 2011 IEEE fifth international conference on semantic computing, pages 368–375. IEEE, 2011.
- [12] Mashael AlSabah, Mohamed Nabeel, Yazan Boshmaf, and Euijin Choo. Content-agnostic detection of phishing domains using certificate transparency and passive dns. In Proceedings of the 25th International Symposium on Research in Attacks, Intrusions and Defenses, pages 446–459, 2022.
- [13] Simone Aonzo, Alessio Merlo, Giulio Tavella, and Yanick Fratantonio. Phishing attacks on modern android. In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, pages 1788–1801, 2018.
- [14] Simon Bell, Kenny Paterson, and Lorenzo Cavallaro. Catch me (on time) if you can: Understanding the effectiveness of twitter url blacklists. arXiv preprint arXiv:1912.02520, 2019.
- [15] Hugo Bijmans, Tim Booij, Anneke Schwedersky, Aria Nedgabat, and Rolf van Wegberg. Catching phishers by their bait: Investigating the dutch phishing landscape through phishing kit detection. In 30th USENIX Security Symposium (USENIX Security 21), pages 3757–3774, 2021.
- [16] Marzieh Bitaab, Haehyun Cho, Adam Oest, Penghui Zhang, Zhibo Sun, Rana Pourmohamad, Doowon Kim, Tiffany Bao, Ruoyu Wang, Yan Shoshitaishvili, et al. Scam pandemic: How attackers exploit public fear through phishing. In 2020 APWG Symposium on Electronic Crime Research (eCrime), pages 1–10. IEEE, 2020.
- [17] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. Advances in neural information processing systems, 33:1877–1901, 2020.
- [18] Lina Brunken, Annalina Buckmann, Jonas Hielscher, and M Angela Sasse. {“To” do this properly, you need more {Resources”}: The hidden costs of introducing simulated phishing campaigns. In 32nd USENIX Security Symposium (USENIX Security 23), pages 4105–4122, 2023.
- [19] Euijin Choo, Mohamed Nabeel, Doowon Kim, Ravindu De Silva, Ting Yu, and Issa Khalil. A large scale study and classification of virustotal reports on phishing and malware urls. Proceedings of the ACM on Measurement and Analysis of Computing Systems, 7(3):1–26, 2023.
- [20] PaddlePaddle Contributors. Paddleocr: Rich multilingual, practical ocr tools, and leading pre-trained models. <https://github.com/PaddlePaddle/PaddleOCR>, 2023.
- [21] Marco Cova, Christopher Kruegel, and Giovanni Vigna. There is no free phish: An analysis of “free” and live phishing kits. WOOT, 8:1–8, 2008.
- [22] Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. arXiv preprint arXiv:2306.06070, 2023.
- [23] Western Digital. Wd my cloud - personal cloud storage. <https://www.westerndigital.com/>

- en-ap/products/cloud-storage/wd-my-cloud#WDBCTL0020HWT-AESN. Accessed on: August 4, 2023.
- [24] Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. A survey for in-context learning. arXiv preprint arXiv:2301.00234, 2022.
- [25] Yongkun Du, Zhineng Chen, Caiyan Jia, Xiaoting Yin, Tianlun Zheng, Chenxia Li, Yuning Du, and Yu-Gang Jiang. Svtr: Scene text recognition with a single visual model. 2022.
- [26] Birhanu Eshete, Adolfo Villafiorita, and Komminist Weldemariam. Binspect: Holistic analysis and detection of malicious web pages. In Security and Privacy in Communication Networks: 8th International ICST Conference, SecureComm 2012, Padua, Italy, September 3-5, 2012. Revised Selected Papers 8, pages 149–166. Springer, 2013.
- [27] Shancheng Fang, Hongtao Xie, Yuxin Wang, Zhendong Mao, and Yongdong Zhang. Abinet: Read like humans: Autonomous, bidirectional and iterative language modeling for scene text recognition. pages 7098–7107, 2021.
- [28] Anthony Y Fu, Liu Wenyin, and Xiaotie Deng. Detecting phishing web pages with visual similarity assessment based on earth mover’s distance (emd). IEEE transactions on dependable and secure computing, 3(4):301–311, 2006.
- [29] Hiroki Furuta, Ofir Nachum, Kuang-Huei Lee, Yutaka Matsuo, Shixiang Shane Gu, and Izzeddin Gur. Multimodal web navigation with instruction-finetuned foundation models. arXiv preprint arXiv:2305.11854, 2023.
- [30] Sujata Garera, Niels Provos, Monica Chew, and Aviel D Rubin. A framework for detection and measurement of phishing attacks. In Proceedings of the 2007 ACM workshop on Recurring malcode, pages 1–8, 2007.
- [31] Hugo Gascon, Steffen Ullrich, Benjamin Stritter, and Konrad Rieck. Reading between the lines: content-agnostic detection of spear-phishing emails. In Research in Attacks, Intrusions, and Defenses: 21st International Symposium, RAID 2018, Heraklion, Crete, Greece, September 10-12, 2018, Proceedings 21, pages 69–91. Springer, 2018.
- [32] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572, 2014.
- [33] Google. Google safe browsing. <https://safebrowsing.google.com/>, 2023. Accessed: 2024.
- [34] Izzeddin Gur, Hiroki Furuta, Austin Huang, Mustafa Safdari, Yutaka Matsuo, Douglas Eck, and Aleksandra Faust. A real-world webagent with planning, long context understanding, and program synthesis. arXiv preprint arXiv:2307.12856, 2023.
- [35] Izzeddin Gur, Ofir Nachum, Yingjie Miao, Mustafa Safdari, Austin Huang, Aakanksha Chowdhery, Sharan Narang, Noah Fiedel, and Aleksandra Faust. Understanding html with large language models. arXiv preprint arXiv:2210.03945, 2022.
- [36] Xiao Han, Nizar Kheir, and Davide Balzarotti. Phish-eye: Live monitoring of sandboxed phishing kits. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, pages 1402–1413, 2016.
- [37] Xuanli He, Yuxiang Wu, Oana-Maria Camburu, Pasquale Minervini, and Pontus Stenetorp. Using natural language explanations to improve robustness of in-context learning for natural language inference. arXiv preprint arXiv:2311.07556, 2023.
- [38] Xiaowei Hu, Zhe Gan, Jianfeng Wang, Zhengyuan Yang, Zicheng Liu, Yumao Lu, and Lijuan Wang. Scaling up vision-language pre-training for image captioning. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 17980–17989, 2022.
- [39] IBM. Cost of a data breach report 2023. Technical report, International Business Machines Corporation (IBM), 2023.
- [40] Luca Invernizzi, Kurt Thomas, Alexandros Kapravelos, Oxana Comanescu, Jean-Michel Picod, and Elie Bursztein. Cloak of visibility: Detecting when machines browse a different web. In 2016 IEEE Symposium on Security and Privacy (SP), pages 743–758. IEEE, 2016.
- [41] Doowon Kim, Haehyun Cho, Yonghwi Kwon, Adam Doupé, Soeul Son, Gail-Joon Ahn, and Tudor Dumitras. Security analysis on practices of certificate authorities in the https phishing ecosystem. In Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security, pages 407–420, 2021.
- [42] Wonjae Kim, Bokyung Son, and Ildoo Kim. Vilt: Vision-and-language transformer without convolution or region supervision. In International Conference on Machine Learning, pages 5583–5594. PMLR, 2021.

- [43] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. ICML Deep Learning Workshop, 2, 2015.
- [44] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. arXiv preprint arXiv:1611.01236, 2016.
- [45] Daniele Lain, Kari Kostiaainen, and Srdjan Čapkun. Phishing in organizations: Findings from a large-scale and long-term study. In 2022 IEEE Symposium on Security and Privacy (SP), pages 842–859. IEEE, 2022.
- [46] Hung Le, Quang Pham, Doyen Sahoo, and Steven CH Hoi. Urlnet: Learning a url representation with deep learning for malicious url detection. arXiv preprint arXiv:1802.03162, 2018.
- [47] Chenliang Li, Haiyang Xu, Junfeng Tian, Wei Wang, Ming Yan, Bin Bi, Jiabo Ye, Hehong Chen, Guohai Xu, Zheng Cao, et al. mplug: Effective and efficient vision-language learning by cross-modal skip-connections. arXiv preprint arXiv:2205.12005, 2022.
- [48] Chunyuan Li, Haotian Liu, Liunian Li, Pengchuan Zhang, Jyoti Aneja, Jianwei Yang, Ping Jin, Houdong Hu, Zicheng Liu, Yong Jae Lee, et al. Elevator: A benchmark and toolkit for evaluating language-augmented visual models. Advances in Neural Information Processing Systems, 35:9287–9301, 2022.
- [49] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. arXiv preprint arXiv:2301.12597, 2023.
- [50] Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. Oscar: Object-semantics aligned pre-training for vision-language tasks. In Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXX 16, pages 121–137. Springer, 2020.
- [51] Yukun Li, Zhenguo Yang, Xu Chen, Huaping Yuan, and Wenyin Liu. A stacking model using url and html features for phishing webpage detection. Future Generation Computer Systems, 94:27–39, 2019.
- [52] Minghui Liao, Zhaoyi Wan, Cong Yao, Kai Chen, and Xiang Bai. Real-time scene text detection with differentiable binarization. In Proceedings of the AAAI conference on artificial intelligence, volume 34, pages 11474–11481, 2020.
- [53] Minghui Liao, Zhisheng Zou, Zhaoyi Wan, Cong Yao, and Xiang Bai. Real-time scene text detection with differentiable binarization and adaptive scale fusion. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022.
- [54] Xu Lin, Panagiotis Ilia, Saumya Solanki, and Jason Polakis. Phish in sheep’s clothing: Exploring the authentication pitfalls of browser fingerprinting. In 31st USENIX Security Symposium (USENIX Security 22), pages 1651–1668, 2022.
- [55] Yun Lin, Ruofan Liu, Dinil Mon Divakaran, Jun Yang Ng, Qing Zhou Chan, Yiwen Lu, Yuxuan Si, Fan Zhang, and Jin Song Dong. Phishpedia: A hybrid deep learning based approach to visually identify phishing webpages. In 30th USENIX Security Symposium (USENIX Security 21), pages 3793–3810, 2021.
- [56] Mingxuan Liu, Yiming Zhang, Baojun Liu, Zhou Li, Haixin Duan, and Donghong Sun. Detecting and characterizing sms spearphishing attacks. In Annual Computer Security Applications Conference, pages 930–943, 2021.
- [57] Ruofan Liu, Yun Lin, Xianglin Yang, Siang Hwee Ng, Dinil Mon Divakaran, and Jin Song Dong. Inferring phishing intention via webpage appearance and dynamics: A deep vision based approach. In 31st USENIX Security Symposium (USENIX Security 22), pages 1633–1650, 2022.
- [58] Ruofan Liu, Yun Lin, Yifan Zhang, Penn Han Lee, and Jin Song Dong. Knowledge expansion and counterfactual interaction for {Reference-Based} phishing detection. In 32nd USENIX Security Symposium (USENIX Security 23), pages 4139–4156, 2023.
- [59] Sheng Lu, Hendrik Schuff, and Iryna Gurevych. How are prompts different in terms of sensitivity? arXiv preprint arXiv:2311.07230, 2023.
- [60] Claudio Marforio, Ramya Jayaram Masti, Claudio Soriente, Kari Kostiaainen, and Srdjan Capkun. Personalized security indicators to detect application phishing attacks in mobile platforms. arXiv preprint arXiv:1502.06824, 2015.
- [61] Sourena Maroofi, Maciej Korczyński, and Andrzej Duda. Are you human? resilience of phishing detection to evasion techniques based on human verification. In Proceedings of the ACM Internet Measurement Conference, pages 78–86, 2020.
- [62] Eric Medvet, Engin Kirda, and Christopher Kruegel. Visual-similarity-based phishing detection. In Proceedings of the 4th international conference on

- Security and privacy in communication networks, pages 1–6, 2008.
- [63] Ron Mokady, Amir Hertz, and Amit H Bermano. Clip-cap: Clip prefix for image captioning. arXiv preprint arXiv:2111.09734, 2021.
- [64] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2574–2582, 2016.
- [65] Adam Oest, Yeganeh Safaei, Adam Doupé, Gail-Joon Ahn, Brad Wardman, and Kevin Tyers. Phishfarm: A scalable framework for measuring the effectiveness of evasion techniques against browser phishing blacklists. In 2019 IEEE Symposium on Security and Privacy (SP), pages 1344–1361. IEEE, 2019.
- [66] Adam Oest, Yeganeh Safaei, Penghui Zhang, Brad Wardman, Kevin Tyers, Yan Shoshitaishvili, and Adam Doupé. {PhishTime}: Continuous longitudinal measurement of the effectiveness of anti-phishing blacklists. In 29th USENIX Security Symposium (USENIX Security 20), pages 379–396, 2020.
- [67] Adam Oest, Yeganeh Safei, Adam Doupé, Gail-Joon Ahn, Brad Wardman, and Gary Warner. Inside a phisher’s mind: Understanding the anti-phishing ecosystem through phishing kit analysis. In 2018 APWG Symposium on Electronic Crime Research (eCrime), pages 1–12. IEEE, 2018.
- [68] Adam Oest, Penghui Zhang, Brad Wardman, Eric Nunes, Jakub Burgis, Ali Zand, Kurt Thomas, Adam Doupé, and Gail-Joon Ahn. Sunrise to sunset: Analyzing the end-to-end life cycle and effectiveness of phishing attacks at scale. In 29th {USENIX} Security Symposium ({USENIX} Security 20), 2020.
- [69] Federal Bureau of Investigation. 2022 internet crime report. Technical report, Federal Bureau of Investigation, 2022.
- [70] OpenAI. Chat completions api. <https://platform.openai.com/docs/guides/gpt/chat-completions-api>, 2023. [Online; accessed 2024].
- [71] Peng Peng, Chao Xu, Luke Quinn, Hang Hu, Bimal Viswanath, and Gang Wang. 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, pages 181–192, 2019.
- [72] Peng Peng, Limin Yang, Linhai Song, and Gang Wang. Opening the blackbox of virustotal: Analyzing online phishing scan engines. In Proceedings of the Internet Measurement Conference, pages 478–485, 2019.
- [73] Fábio Perez and Ian Ribeiro. Ignore previous prompt: Attack techniques for language models. arXiv preprint arXiv:2211.09527, 2022.
- [74] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In International conference on machine learning, pages 8748–8763. PMLR, 2021.
- [75] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. arXiv preprint arXiv:1710.05941, 2017.
- [76] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. Advances in neural information processing systems, 28, 2015.
- [77] Antonio Ruggia, Andrea Possemato, Alessio Merlo, Dario Nisi, and Simone Aonzo. Android, notify me when it is time to go phishing. In EUROS&P 2023, 8th IEEE European Symposium on Security and Privacy, 2023.
- [78] Iskander Sanchez-Rola, Leyla Bilge, Davide Balzarotti, Armin Buescher, and Petros Efstathopoulos. Rods with laser beams: understanding browser fingerprinting on phishing pages. In 32nd USENIX Security Symposium (USENIX Security 23), pages 4157–4173, 2023.
- [79] Steve Sheng, Brad Wardman, Gary Warner, Lorrie Cranor, Jason Hong, and Chengshan Zhang. An empirical analysis of phishing blacklists. 2009.
- [80] Baoguang Shi, Xiang Bai, and Cong Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(11):2298–2304, 2017.
- [81] Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, et al. Language models are multilingual chain-of-thought reasoners. arXiv preprint arXiv:2210.03057, 2022.
- [82] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In Advances in neural information processing systems, 2017.

- [83] Gianluca Stringhini and Olivier Thonnard. That ain't you: Blocking spearphishing through behavioral modelling. In International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment, pages 78–97. Springer, 2015.
- [84] Karthika Subramani, William Melicher, Oleksii Starov, Phani Vadrevu, and Roberto Perdisci. Phishinpatterns: measuring elicited user interactions at scale on phishing websites. In Proceedings of the 22nd ACM Internet Measurement Conference, pages 589–604, 2022.
- [85] Kurt Thomas, Frank Li, Ali Zand, Jacob Barrett, Juri Ranieri, Luca Invernizzi, Yarik Markov, Oxana Comanescu, Vijay Eranti, Angelika Moscicki, et al. Data breaches, phishing, or malware? understanding the risks of stolen credentials. In Proceedings of the 2017 ACM SIGSAC conference on computer and communications security, pages 1421–1434, 2017.
- [86] Ke Tian, Steve TK Jan, Hang Hu, Danfeng Yao, and Gang Wang. Needle in a haystack: Tracking down elite phishing domains in the wild. In Proceedings of the Internet Measurement Conference 2018, pages 429–442, 2018.
- [87] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. arXiv preprint arXiv:2307.09288, 2023.
- [88] Certificate Transparency. Certificate transparency. <https://certificate.transparency.dev/>, 2023. [Online; accessed 2023].
- [89] Güliz Seray Tuncay, Jingyu Qian, and Carl A Gunter. See no evil: phishing for permissions with false transparency. In 29th USENIX Security Symposium (USENIX Security 20), pages 415–432, 2020.
- [90] Amber Van Der Heijden and Luca Allodi. Cognitive triaging of phishing attacks. In 28th USENIX Security Symposium (USENIX Security 19), pages 1309–1326, 2019.
- [91] Javier Vargas, Alejandro Correa Bahnsen, Sergio Villegas, and Daniel Ingevaldson. Knowing your enemies: Leveraging data analysis to expose phishing patterns against a major us financial institution. In 2016 APWG Symposium on Electronic Crime Research (eCrime), pages 1–10. IEEE, 2016.
- [92] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. Advances in neural information processing systems, 30, 2017.
- [93] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, and Daan Wierstra. Matching networks for one shot learning. In Advances in neural information processing systems, 2016.
- [94] Boshi Wang, Sewon Min, Xiang Deng, Jiaming Shen, You Wu, Luke Zettlemoyer, and Huan Sun. Towards understanding chain-of-thought prompting: An empirical study of what matters. arXiv preprint arXiv:2212.10001, 2022.
- [95] David Y Wang, Stefan Savage, and Geoffrey M Voelker. Cloak and dagger: dynamics of web search cloaking. In Proceedings of the 18th ACM conference on Computer and communications security, pages 477–490, 2011.
- [96] Xinyi Wang, Wanrong Zhu, Michael Saxon, Mark Steyvers, and William Yang Wang. Large language models are latent variable models: Explaining and finding good demonstrations for in-context learning. In Thirty-seventh Conference on Neural Information Processing Systems, 2023.
- [97] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. Advances in Neural Information Processing Systems, 35:24824–24837, 2022.
- [98] Guang Xiang, Jason Hong, Carolyn P Rose, and Lorie Cranor. Cantina+ a feature-rich machine learning framework for detecting phishing web sites. ACM Transactions on Information and System Security (TISSEC), 14(2):1–28, 2011.
- [99] Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit bayesian inference. arXiv preprint arXiv:2111.02080, 2021.
- [100] Pengchuan Zhang, Xiujun Li, Xiaowei Hu, Jianwei Yang, Lei Zhang, Lijuan Wang, Yejin Choi, and Jianfeng Gao. Vinvl: Revisiting visual representations in vision-language models. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 5579–5588, 2021.
- [101] Penghui Zhang, Adam Oest, Haehyun Cho, Zhibo Sun, RC Johnson, Brad Wardman, Shaown Sarker, Alexandros Kapravelos, Tiffany Bao, Ruoyu Wang, et al. Crawlphish: Large-scale analysis of client-side cloaking techniques in phishing. In 2021 IEEE Symposium on Security and Privacy (SP), pages 1109–1124. IEEE, 2021.

- [102] Penghui Zhang, Zhibo Sun, Sukwha Kyung, Hans Walter Behrens, Zion Leonahenahe Basque, Haehyun Cho, Adam Oest, Ruoyu Wang, Tiffany Bao, Yan Shoshitaishvili, et al. 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, pages 3165–3179, 2022.
- [103] Yue Zhang, Jason I Hong, and Lorrie F Cranor. Cantina: a content-based approach to detecting phishing web sites. In Proceedings of the 16th international conference on World Wide Web, pages 639–648, 2007.
- [104] Xinyu Zhou, Cong Yao, He Wen, Yuzhi Wang, Shuchang Zhou, Weiran He, and Jiajun Liang. East: an efficient and accurate scene text detector. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pages 5551–5560, 2017.
- [105] Shuofei Zhu, Jianjun Shi, Limin Yang, Boqin Qin, Ziyi Zhang, Linhai Song, and Gang Wang. Measuring and modeling the label dynamics of online {Anti-Malware} engines. In 29th USENIX Security Symposium (USENIX Security 20), pages 2361–2378, 2020.

A Appendix

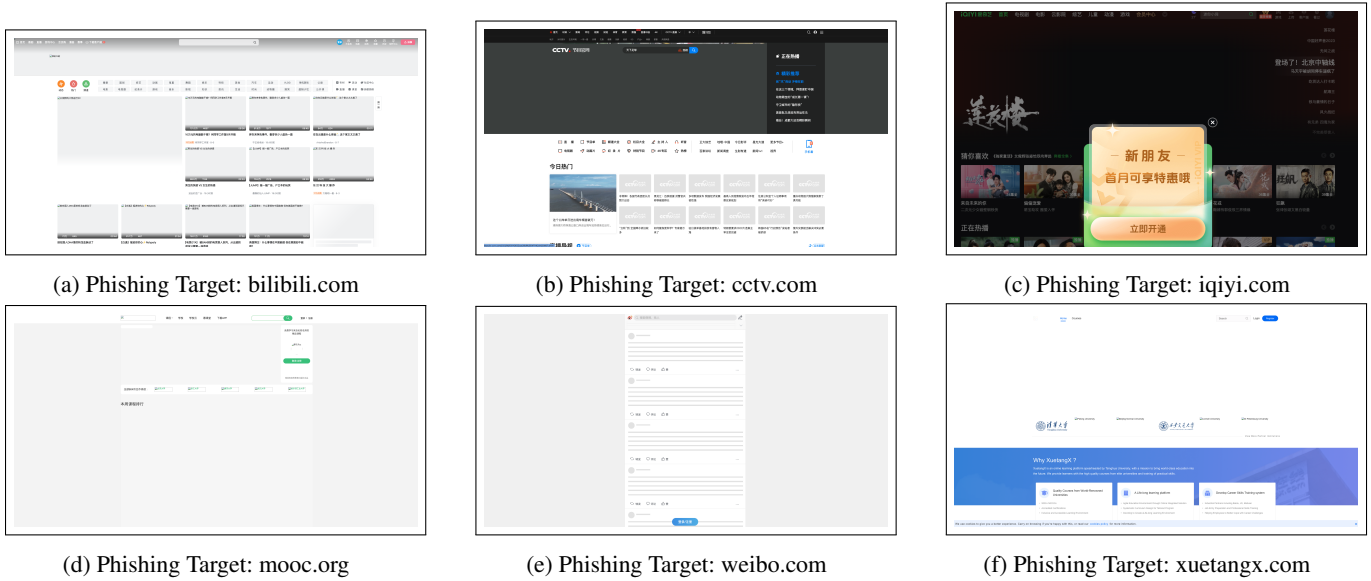


Figure 13: The phishing site <https://x.xsteach.cf/> randomly displays one phishing page out of the six pages.

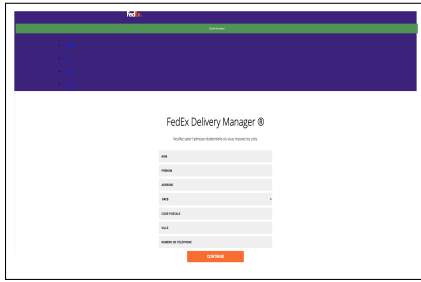
A.1 Prompt Comparison

In this section, we present qualitative examples of LLM’s predictions both with and without adherence to the design principle of minimal entropy. Briefly, in the absence of this restriction, LLM tends to generate rambling and random responses.

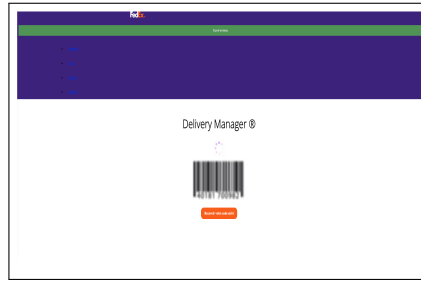
Regarding the CRP prediction model, our objective is the opposite; we seek a more extended response that includes a detailed reasoning process. Without any specification, the model would only output ‘A’ or ‘B’, without any explanation.

A.2 Field Study for DynaPhish [58]

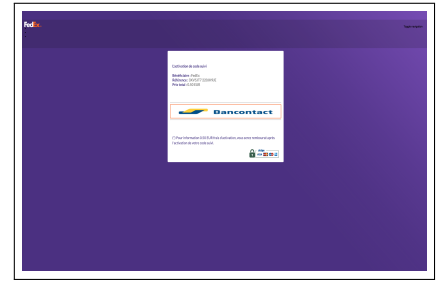
Due to the costs associated with using the Google Cloud API and Google Search API, a large-scale comparison involving DynaPhish was not feasible. Instead, we conducted a small-scale field study for comparative analysis. This study spanned five days, from January 22 to January 29, 2024, with the same setup as our previous one-month field study. During this period, we crawled about 15,000 emerging websites from Certstream and annotated the reported phishing instances by DynaPhish, *PhishLLM*, *Phishpedia*, and *PhishIntention*, subsequently calculating their precision.



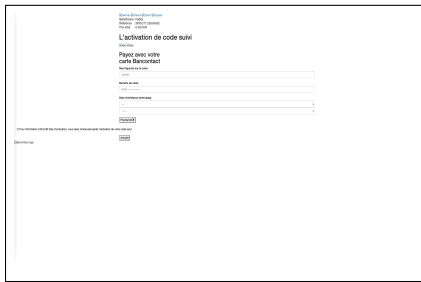
(a) Landing page



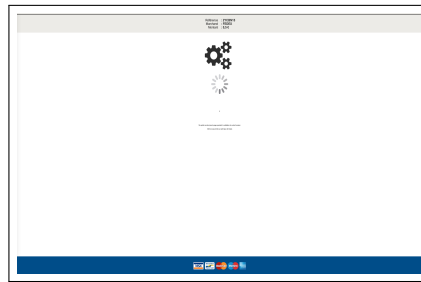
(b) Loading page



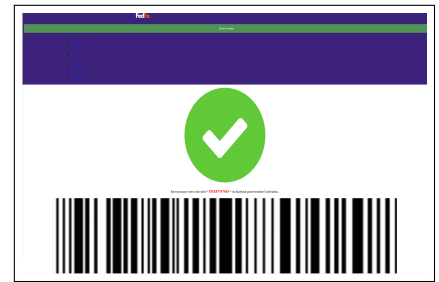
(c) Payment page



(d) Payment page

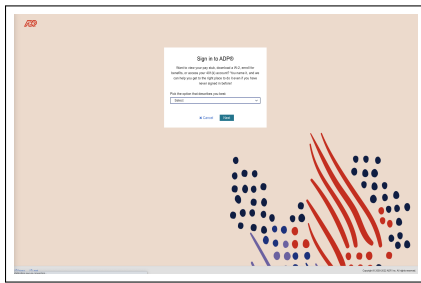


(e) Loading page



(f) Confirmation page

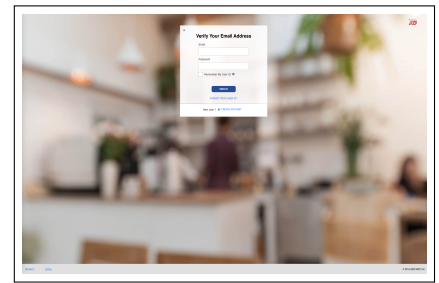
Figure 14: An example of extensive data collection: <https://votre-formulaireliv.com/>



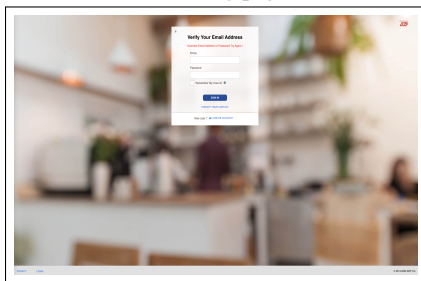
(a) Landing page



(b) UserID page



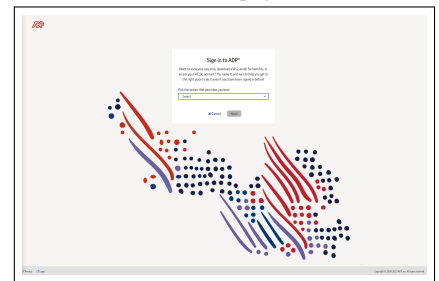
(c) Email page



(d) Fake error page



(e) Loading page



(f) Redirect to login.adp.com/welcome

Figure 15: An example of redirection to target site: <https://cxeradp.tech>

Table 8: Comparison of different prompt design for brand recognition Task, GPT’s responses are highlighted in yellow. Note that, a prompt without any restriction will produce lengthy and ambiguous outputs. The text with red, green, and purple background is an abstract phrase for concrete task background, answer instruction, and few-shot example.

Regular Prompt	Minimal Entropy-Based Prompt
<p>Task background + Given the following description on the brand’s logo: “a white background with a blue text that reads image twist”, the logo’s OCR text: “IMageTWIST not just another image hosting”. Question: What is the brand’s domain? Answer:</p>	<p>Task background + Answer instruction + Few-shot examples + Given the following description on the brand’s logo: “a white background with a blue text that reads image twist”, the logo’s OCR text: “IMageTWIST not just another image hosting”. Question: What is the brand’s domain? Answer:</p>
The brand’s domain is likely related to image hosting or image sharing	imagetwist.com
<p>Task background + Given the following description on the brand’s logo: “a red and white logo with the words hetzer”, the logo’s OCR text: “HETZNER”. Question: What is the brand’s domain? Answer:</p>	<p>Task background + Answer instruction + Few-shot examples + Given the following description on the brand’s logo Question: What is the brand’s domain? Answer:</p>
The brand’s domain is likely related to web hosting or data centers. Hetzner is a well-known German web hosting company	hetzner.com
<p>Task background + Given the following description on the brand’s logo: “the educba logo on a black background”, the logo’s OCR text: “EDUCBA”. Question: What is the brand’s domain? Answer:</p>	<p>Task background + Answer instruction + Few-shot examples + Given the following description on the brand’s logo: “the educba logo on a black background”, the logo’s OCR text: “EDUCBA”. Question: What is the brand’s domain? Answer:</p>
The brand’s domain is likely in the education sector	educba.com

```

1 var obfuscate_button = function() {
2   let returned_buttons = document.getElementsByTagName("button"); // get all <button>
3   for (let button of returned_buttons) {
4     var buttonLink = button.getAttribute('href');
5     html2canvas(button).then(function(canvas) {
6       // Create a new image element
7       var img = document.createElement('img');
8       img.src = canvas.toDataURL();
9       var a = document.createElement('a');
10      a.href = buttonLink;
11      a.appendChild(img);
12      button.parentNode.replaceChild(a, button);
13    });
14  }
15 }

```

Figure 16: Javascript code for performing HTML obfuscation attack

Table 9: Comparison of Different Prompt Design for CRP Prediction Task, GPT’s responses are highlighted in yellow. Note that, a prompt without any restriction will produce decisions with no explanation. The text with red, green, and purple background is an abstract phrase for concrete task background, answer instruction, and CoT based few-shot examples.

Regular Prompt	Chain-of-Thought Prompt
<p>Task background + Given the webpage content: “ebay Hi Sign in to eBay or create an account Email or username Created your account with a mobile number? Sign in with mobile Continue”, Question: A. This is a credential-requiring page. B. This is not a credential-requiring page. Answer:</p> <p>A. This is a credential-requiring page</p>	<p>Task background + Answer instruction + CoT based few-shot examples + Given the webpage content: “ebay Hi Sign in to eBay or create an account Email or username Created your account with a mobile number? Sign in with mobile Continue”, Question: A. This is a credential-requiring page. B. This is not a credential-requiring page. Answer:</p> <p>First we filter the keywords that are related to sensitive information: Email, username, mobile number. Then we find the keywords that are related to login: Sign in. Therefore the answer would be A</p>
<p>Task background + Given the webpage content: “LIPUTAN6 berita apa yang ingin anda baca hari ini? CARI MASUK HOME NEWS PEMILU BISNIS BOLA CRYPTO SHOWBIZ TEKNO FOTO HOT CEK FAKTA ISLAMI CITIZEN6 LAINNYA v Masuk Gunakan akun media sosial Anda untuk dapat mengakses seluruh fitur Liputan6.com.”, Question: A. This is a credential-requiring page. B. This is not a credential-requiring page. Answer:</p> <p>A. This is a credential-requiring page</p>	<p>Task background + Answer instruction + CoT based few-shot examples + Given the webpage content: “LIPUTAN6 berita apa yang ingin anda baca hari ini? CARI MASUK HOME NEWS PEMILU BISNIS BOLA CRYPTO SHOWBIZ TEKNO FOTO HOT CEK FAKTA ISLAMI CITIZEN6 LAINNYA v Masuk Gunakan akun media sosial Anda untuk dapat mengakses seluruh fitur Liputan6.com.”, Question: A. This is a credential-requiring page. B. This is not a credential-requiring page. Answer:</p> <p>First we filter the keywords that are related to sensitive information: Email, Nama Pengguna, Kata Sandi. Then we find the keywords that are related to login: Masuk, Gunakan akun media sosial Anda. Therefore the answer would be A</p>
<p>Task background + Given the webpage content: “M indiamart Shopping Sign In v Search for products find verified sellers near you 9 All India X Q Search Sign In Email ID Enter your Email ID ”, Question: A. This is a credential-requiring page. B. This is not a credential-requiring page. Answer:</p> <p>B. This is not a credential-requiring page</p>	<p>Task background + Answer instruction + CoT based few-shot examples + Given the webpage content: “M indiamart Shopping Sign In v Search for products find verified sellers near you 9 All India X Q Search Sign In Email ID Enter your Email ID ”, Question: A. This is a credential-requiring page. B. This is not a credential-requiring page. Answer:</p> <p>First we filter the keywords that are related to sensitive information: Email ID, Enter your Email ID. Then we find the keywords that are related to login: Sign In. Therefore the answer would be A</p>

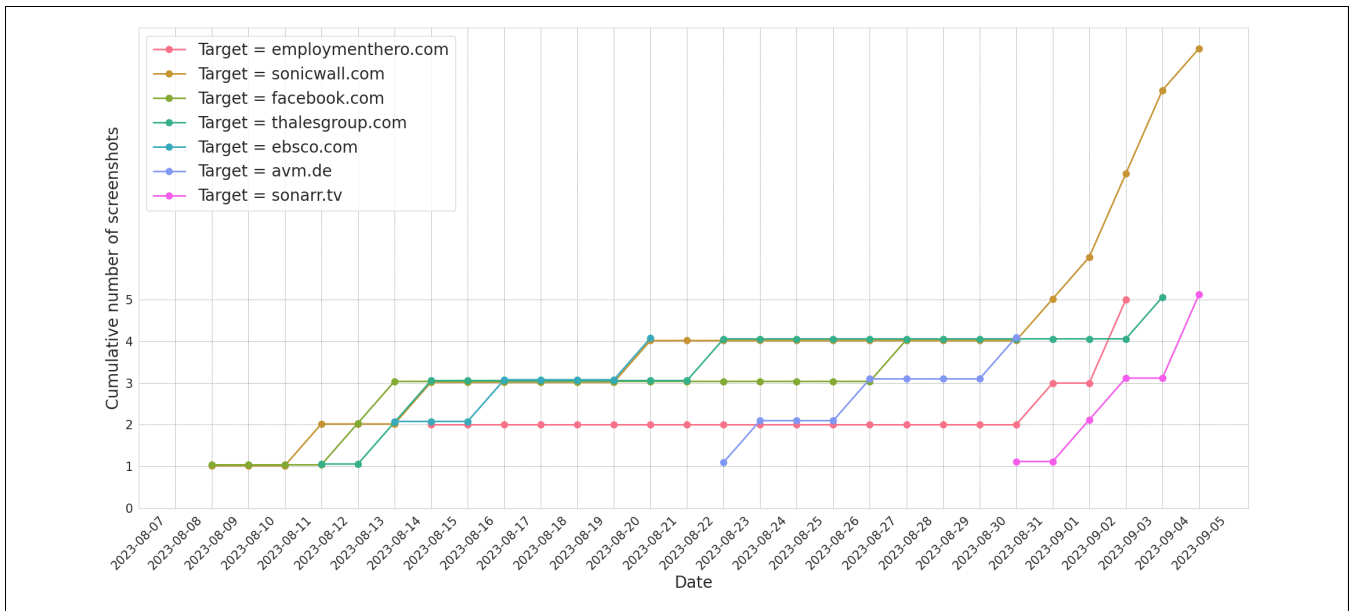


Figure 17: Phishing campaigns over time

(a) Logo implementation of facebook.com

```

1 

```

(b) Logo implementation of .click Facebook phishing

```

1 <div class="image-background p-absolute"></div>
2 <!-- .... -->
3 <script type="text/javascript" id="event_data">
4   <!-- .... -->
5   "44trowte": {
6     "type": "image-block",
7     "specials": {
8       "imageCompression": true,
9       "isTransparent": true,
10      "src": "https://statics.pancake.vn/web-media/e6/9f/8d/0e/838453201
11          ↪ b6e459e07bfb1cca5f976083f0d46836fb6f43696787518.png"
12    },
13    "id": "44trowte",
14    "$parent": "k5qz248d"
15  }
16  <!-- .... -->
17 </script>

```

Figure 18: Meta phishing campaign: Logo implementation comparison

(a) Input implementation of facebook.com

```
1 <input id="email" type="text" class="inputtext _55r1 inputtext _1kbt inputtext _1kbt" placeholder="Email or
  ↳ mobile number" name="email" tabindex="0" value="" autofocus="1" autocomplete="username" aria-
  ↳ label="Email or mobile number">
2 <input id="pass" type="password" class="inputtext _55r1 inputtext _9npi inputtext _9npi" placeholder="
  ↳ password" name="pass" tabindex="0" value="" autocomplete="current-password" aria-label="password">
```

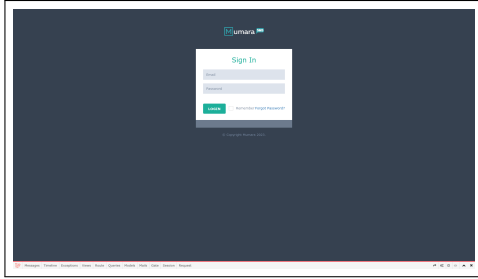
(b) Input implementation of .click Facebook phishing

```
1 <input id="wi-rkphpjrf" type="text" class="full-width full-height" placeholder="Email or phone number"
  ↳ name="full_name" required="">
2 <input id="wi-a7b44eng" type="text" class="full-width full-height" placeholder="Password" name="
  ↳ text_input_1" minlength="6" required="">
```

Figure 19: Meta phishing campaign: Input implementation comparison

```
1 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
2 <html><head>
3 <!-- .... -->
4 <meta name="SonicWall Administrator" content="Copyright 2004 (c) SonicWall, Inc. All rights reserved.">
5 <title>SonicWall - Authentication</title>
6 <meta name="id" content="auth">
7 </head>
8 <frameset name="frameMain" id="frameMain" rows="100%,0%,0%">
9   <frame src="auth1.html" name="authFrm" id="authFrm" scrolling="auto">
10  <frame src="policyBanner.html" name="policyBannerFrm" id="policyBannerFrm" scrolling="no">
11  <frame src="emptyView4.html" name="authTgtFrm" id="authTgtFrm" scrolling="no">
12 </frameset>
13 </html>
```

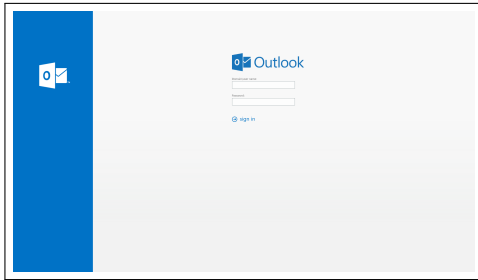
Figure 20: SonicWall phishing campaign: Embedded frame



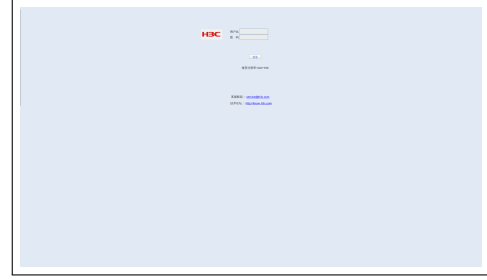
smsplus9.azeem.dev.jt.hostingshouse.com
(a) Phishing Target: mumara.com



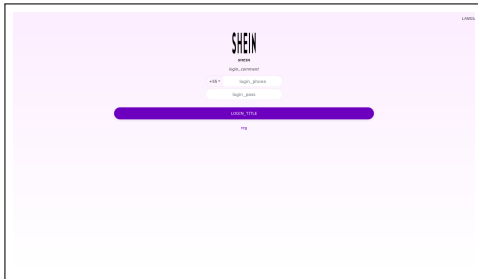
swedbank.pieēja-lv.info
(b) Phishing Target: swedbank.com



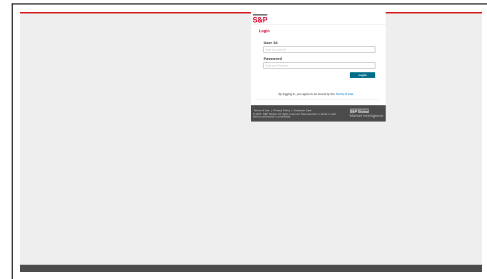
device-8b248998-6847-4def-9eb1-9b59fb283b04.remotewd.com
(c) Phishing Target: outlook.com



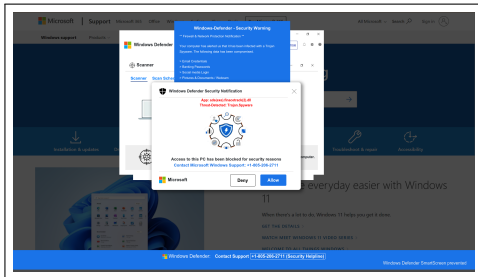
0a35f634d33ec266d03c0fb6f9c377be09d46ab0.ops.yunaq.com
(d) Phishing Target: h3c.com



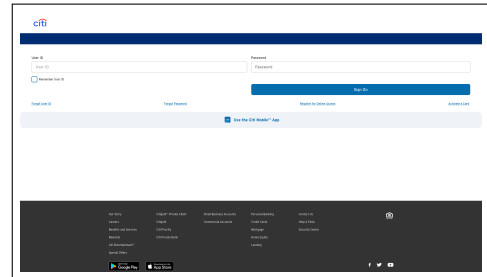
dsgbb.top
(e) Phishing Target: shein.com



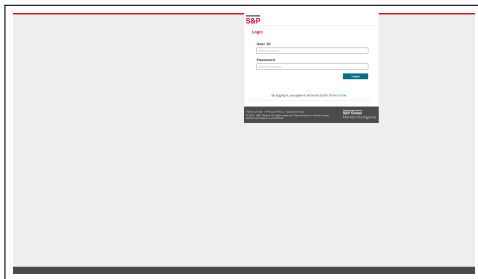
ib-trans-ateam9.sup.issuebook-np.ihsmarkit.com
(f) Phishing Target: spglobal.com



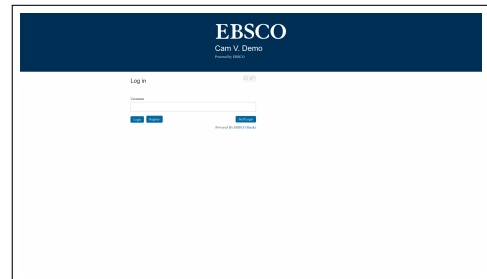
www.office.drivprint.com
(g) Phishing Target: microsoft.com



onlinemngmtciti.com
(h) Phishing Target: citi.com



ib-trans-ateam9.sup.issuebook-np.ihsmarkit.com
(i) Phishing Target: spglobal.com



camvdemo.stackspatform.com
(j) Phishing Target: ebSCO.com

Figure 21: Real Phishing Examples