Email Cloaking: Deceiving Users and Spam Email Detectors with Invisible HTML Settings

Bingyang Guo[1⋆], Mingxuan Liu[2*], Yihui Ma[3], Ruixuan Li[3], Fan Shi[1], Min Zhang[1], Baojun Liu[2,3], Chengxi Xu[1], Haixin Duan[2,3], Geng Hong[4], Min Yang[4], and Qingfeng Pan[5]

[1] National University of Defense Technology, Hefei, China
[2] Zhongguancun Laboratory, Beijing, China
[3] Tsinghua University, Beijing, China
[4] Fudan University, Shanghai, China
[5] Coremail, Guangzhou, China

Abstract. Development of HTML emails increases parsing complexity and discrepancies. Owing to parsing and rendering differences, email systems expose a new attack surface: Cloaked Spam Email (CSE). CSE exploits the legitimate functions of HTML and Cascading Style Sheets (CSS) to build invisible content for cloaking. It can stealthily bypass spam engines and deceive users. However, there is a lack of the understanding of this novel email cloaking threat, let alone a systematic assessment of its threat impacts, leaving a defense gap.

To fill the understanding gap of CSE risk, this paper reveals its threat impacts via empirical analysis and real-world measurements. First, through systematic analysis of CSS rendering features and their applicability to email clients, we identified 16 invisible configurations. Based on these findings, we conducted a comprehensive evaluation of 14 well-known email services. Our results reveal 12 services vulnerable to CSE, with our constructed spam samples successfully bypassing their detection and reaching victim inboxes, including Gmail, Fastmail, and QQ. To systematically assess the impact of CSEs in the wild, we developed a detection framework and applied it to two real-world spam datasets: an open-source spam dataset and the actual logs from a renowned email service provider. Through analyzing a combined total of 8,816,785 emails, we successfully detected 102,156 CSE attacks, highlighting the presence of such threats in the email ecosystem. Finally, we responsibly disclosed these vulnerabilities to affected email providers and provided mitigation recommendations against CSE threat.

Keywords: Abusive Content · Email Cloaking · Spam Detection Bypass

1 Introduction

Email services have become one of the most popular communication platforms, attracting numerous spammers to disseminate deceptive content via email ser-

⋆ Both are first authors.
Corresponding author: zhangmindy@nudt.edu.cn and lbj@tsinghua.edu.cn

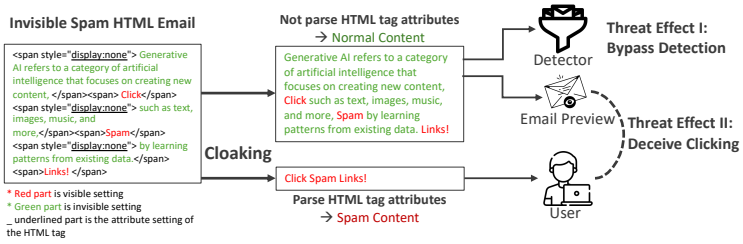


Fig. 1: Threat example of Cloaked Spam Email (CSE).

vices. Statistics indicate that the average number of spam emails sent per day is 85 billion [46], resulting in potential financial losses of up to 1.026 trillion dollars [16]. Spam filtering relies on multiple factors, including sender verification (e.g., SPF) [27] and content detection [7, 19, 20, 38]. Given spam's need for malicious content, content-based detection is the primary defense against it [17, 47]. However, the development of HTML emails has brought content-parsing issues, creating new attack surfaces for traditional content-based detection methods.

We uncovered a novel email cloaking technique, which is termed as **Cloaked Spam Email (CSE)**. CSE can covertly bypass spam detection engines. As shown in Figure 1, it utilizes the CSS configuration in HTML emails that can create invisible effects, inserting interfering text content into the emails. This interferes with the judgment of spam detection engines, thus achieving the goal of bypassing detection. Specifically, since clients parse and render emails strictly according to the configuration for the purpose of displaying them, the content embedded by attackers is invisible in the recipient's email client. However, as detection engines do not need to display emails, they will read all the text content without any rendering, making the embedded content visible to spam detection engines. These embedded texts (usually legitimate) change the semantic meaning of spam emails, enabling them to stealthily bypass spam detection engines. Moreover, CSE can also increase the likelihood of users clicking on spam emails. As shown in Figure 2, attackers utilize the differences between the email preview interface and the main text interface. By using invisible tags, they embed normal content at the beginning of the email, ensuring that normal email content is seen in the email summary, thereby inducing users to click on spam emails.

Research Gap. Significant efforts have been made to enhance the security of email services, such as SPF [27], DKIM [2], and DMARC [32], which verify the integrity and confidentiality of senders and emails, numerous studies have investigated efficient spam detection based on email headers [30], sending behavior [24], and email content [7, 19, 20, 38]. However, these defense mechanisms have not recognized the impact of email cloaking techniques that utilize invisible attribute CSS properties. Although Betts et al [5] recognized the potential risks of text content hiding, it did not conduct a systematic analysis of email cloaking, thus still lacking a systematical understanding of novel CSE threat. Therefore, a

comprehensive assessment of CSE risks is crucial for enhancing email providers' defense capabilities against this stealthy bypassing action.

Threat Impact of CSE. To assess CSE threat, we conducted systematic tests on 14 major email providers. First, by thoroughly analyzing HTML implementation standards [6] and validating them in email clients, we identified 16 invisible configurations that can render text content invisible. We categorized them into five groups: 1) visibility control; 2) color-related; 3) size-related; 4) layout-related; and 5) content manipulation. To test whether the use of invisible CSS configurations could bypass spam detection engines, we collected test samples from open-source spam datasets and selected 14 email providers. We embedded 16 invisible CSS properties within these spam samples separately to interfere with the detection systems' judgment of email content while ensuring user readability, resulting in a total of 2,240 test samples. Finally, we sent these spam test samples to the controlled test accounts of the 14 email providers (ensuring no impact on regular users) and recorded whether these emails were identified as spam. The results revealed that 12 email providers, e.g., Gmail and iCloud, were vulnerable to CSE risks, with test samples successfully bypassing detection engines and reaching users' inboxes.

Real-world Impact of CSE. Additionally, we conducted a large-scale assessment of CSE impact using two real-world spam datasets: an open-source spam dataset and the actual logs from a leading email service provider. Specifically, we developed CSEMiner for identifying CSE and applied it to the two real-world spam datasets. Through analyzing a total of 8,816,785 emails, CSEMiner successfully identified 102,156 CSE attacks, involving 73,202 distinct spam senders. In addition, among the 170,260 suspicious emails provided by the partner email providers, we found that 8,561 (5.03%) employed CSE techniques, from which we identified 324 distinct spam campaigns.

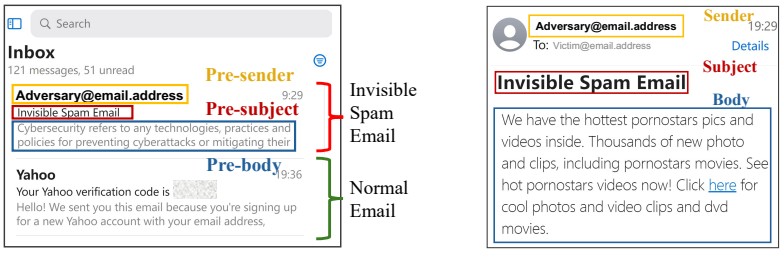
Contributions. The paper makes the following contributions:

• We systematically unveil a novel email cloaking technique, termed Cloaked Spam Email (CSE), which exploits invisible CSS property configurations. CSE not only circumvents spam detection mechanisms but also enhances the deceptive impact on users. Our evaluation code and partial results are open-sourced at https://github.com/MingxuanLiu/Cloaked_Spam_Email-ESORICS25.

• We comprehensively evaluate the threat impact of CSE on 14 mainstream email providers. Our threat assessment reveals that 12 email providers are vulnerable to CSE threat. We have responsibly disclosed this risk to all affected vendors, receiving positive responses.

• Threat impact measurements carried out on two real-world datasets reveal that Cloaked Spam Email (CSE) technology has been misused to create spam emails. Empirical assessments indicate that these techniques pose a threat to the current email systems.

[6] https://html.spec.whatwg.org/multipage/



(a) Email Preview Interface. (b) Full Email View of CSE.

Fig. 2: Example of email preview interface and full email view.

2 Background and Threat Model of CSE

In this section, we illustrate the preliminary information of email service and the detailed threat description of CSE.

2.1 Background

Email services, utilizing the Simple Mail Transfer Protocol (SMTP) [37], operate through a multi-step process: 1) Sender's mail user agent (MUA) transmits email to mail transfer agent (MTA) via SMTP or HTTP; 2) Sender's MTA forwards to receiver's MTA via SMTP; 3) Receiver's MTA delivers to receiver's MUA via HTTP, IMAP, or POP3 [29]; 4) Receiver's MUA parses and displays the email content to users. Driven by demand for more complex content presentation (e.g., image), email communication evolved from simple plain text in the early internet era to HTML-based formats [13]. Similar to HTML parsing and rendering of websites, email terminals must adhere to various HTML configurations, such as font and font size, when displaying emails on the User Interface (UI). However, email clients' limited HTML rendering capabilities and security concerns led to the adoption of a restricted HTML subset for emails [34]. Specifically, HTML email excludes certain HTML tags (like `<iframe>`) and JavaScript (JS) support, balancing enhanced visual presentation with compatibility and security considerations across diverse email platforms.

As HTML format is introduced into email content, the rendering and display of email clients have become increasingly complex, also introducing some rendering discrepancies between different components. This parsing difference mainly stems from the functional disparities among different components, that is, those components that require UI display and those that do not. For example, the following two sets of parsing differences.

Parsing difference I: between spam detector and full email view for users. The email content is a key feature for spam detection [40], so many vendors' detectors aim for comprehensive content processing. Additionally, due to the lack of complex UI display requirements and performance limitations in handling large volumes of emails, spam detectors typically do not render emails

but instead extract the content of interest from semi-structured email files for analysis. Specifically, they usually directly extract all the text content within emails, not performing any rendering process. This leads to a difference where detectors can see text set as invisible, while users cannot view that portion from the email UI interface (i.e., full email view).

Parsing difference II: between preview interface and full email view. The email preview is the first UI displayed to users in the MUA (as shown in Figure 2a). Users can only see the complete email content after clicking on a specific email (as shown in Figure 2b). The email preview typically consists of three parts: pre-sender (sender name or email address), pre-subject (email subject) and pre-body (email summary). The purpose of the email preview is to provide users with a summary before opening the email, significantly enhancing processing efficiency. The pre-body reflects the summary of the full email content, which is typically a limited number of initial characters as the summary (e.g., Gmail allows summaries from 0 to 5 lines) [12]. Since pre-body only displays text, lacking strong rendering requirements, and thus often does not adhere to CSS properties. This results in a rendering discrepancy between the pre-body and body, where the pre-body can show text set as invisible, but the body cannot.

2.2 Threat Exploitation of CSE

CSE exploits the rendering discrepancies to create email cloaking threat, allowing it to bypass spam detection (threat effect I), but also deceiving users into clicking on spam emails (threat effect II).

Threat Effect I: Bypass spam detection. Parsing difference I of email content seen between spam detectors and users' full email view allows CSE to evade detection. Because spam detectors do not render CSS properties, they process both invisible and visible text together. As shown in Figure 1, adversaries embed "normal" distracting text within CSS properties with invisible attributes. This distracting text truncates the malicious content in spam, undermining its original harmful semantics and making it difficult for spam detectors to identify malicious features, thereby bypassing detection. However, since users see content rendered according to HTML attributes, they do not view the "normal" text used for truncation, allowing the carefully crafted spam content to be fully conveyed to the victim, while bypassing detection, as shown in Figure 1.

Threat Effect II: Deceive victim clicking. Parsing difference II between the pre-body in the email preview and the body in the full view allows CSE to deceive victims into clicking on spam. Since the pre-body displays only email content, it does not strictly adhere to HTML rendering. Due to the space-constrained, preview interface lacking complex-format display needs, pre-body usually neither needs nor adheres to HTML configurations when extracting text. Adversaries embed "normal" content at the beginning of the email, ensuring that the pre-body shows legitimate content to users, thereby reducing victims' suspicion towards CSE emails and increasing the likelihood of clicks, as shown in Figure 2.

Threat Model. CSE has two main objectives. To ensure the success of spam email attacks, adversaries need to ensure that the emails can successfully reach

the recipients' inboxes, that is, bypass spam detection engines. Regarding enhancing the success rate of spam email attacks, adversaries need to deceive victims into opening the emails in the email preview interface as much as possible. Implementing CSE is straightforward for adversaries, as they need to meet two conditions to attempt sending CSE to any victim's email and evade spam detection: 1) a server or email account capable of sending emails without being blocked by the target email system; 2) knowledge of CSS properties with invisible configurations. Victims can be threatened by CSE as long as they have an email account for receiving emails and a MUA for checking them.

3 Evaluation of CSE Threat Impact

To evaluate the threat impact of CSE, we conducted tests on popular email services. Initialized by the selection of target email services, we first systematically examined CSS properties which could cause invisible parsing effect. Then, we tested the target email services by constructing a series of test samples with invisible settings to evaluate whether they could bypass spam detection engines.

3.1 Selection of Target Email Services

To conduct the tests, we need accounts from target email service providers. Based on market popularity, we selected dominant email service providers, most of which have over 1 billion users. Among them, we chose 25 email service providers that allow public registration, which were also testing targets in prior studies [22, 39].

Considering that the attack goal of CSE is primarily to bypass the spam detection engine, it is necessary to simultaneously identify a batch of spam samples and email providers, ensuring that spam samples without CSE technology can be stably identified as spam by the spam detection engines of email providers. We conducted a pre-experiment based on the open-source spam email datasets (detailed in Section 3.3). Specifically, we randomly sampled 60 spam emails from these three datasets and sent plain spam text (without hidden configurations) to test email accounts of selected providers, observing whether the emails reached the spam folders of their users. Table 7 shows the results. Four email providers produced inconsistent results in two consecutive tests on the same spam sample, making it difficult to determine their performance to detect the test sample. Seven email providers only classified a few (fewer than 10) spam emails, including Tutamail.com and Rambler.ru, and Five providers did not classify any test emails as spam (e.g., Naver.com). Finally, we ultimately selected 14 email providers with stable detection performance in tests of 60 spam emails—those showing no inconsistent results and identifying at least 10 spam emails.

3.2 Identification of Invisible CSS Properties

To comprehensively identify CSS properties with invisible configurations suitable for constructing CSE, we employed a combination of static and dynamic analysis.

Table 1: CSS properties and invisible configurations.

Category	Invisible Configurations
Visibility control	(A1) display: none;
	(A2) visibility: hidden;
	(A3) visibility: collapse;
Color control	(B1) color: transparent;
	(B2) opacity: x^*;
	(B3) blur: x^*;
	(B4) text-to-background contrast: $y^{\#}$;
	(B5) mix-blend-mode: x^*
Size control	(C) font-size : x^*;
Layout control	(D1) position: absolute; margin/padding: x^*
	(D2) position: absolute; left/top/right/bottom: x^*
	(D3) transform: translate[X|Y] (x^*)
	(D4) line-height/text-indent:x^*
Content Manipulation	(E1) position & z-index
	(E2) mask/-webkit-mask
	(E3) clip: rect(x^*) / clip-path

x^*: A constructed value based on the effect of the corresponding CSS property.

$y^{\#}$: The degree of difference between font color and background color.

Static analysis. First, we identify CSS properties with invisible potential from statically analyzing standard documents, as they may pose CSE risks. Specifically, two security researchers manually reviewed HTML standards [7], identifying 16 CSS properties capable of creating invisible effects and categorizing them into five strategic groups. Table 1 summarizes the principles and implementations of these 5 classes of CSS properties.

• *Visibility control.* To enhance dynamic control and rendering performance, HTML provides 3 CSS properties that can directly hide elements on the screen, such as "(A1) `display:none`".

• *Color Control.* To enhance visual richness in emails, HTML allows the adjustment of opacity and color of text and background. On one hand, text transparency makes it invisible. On the other hand, a low text-background color contrast makes text blend into the background, achieving an invisible effect.

• *Size Control.* In HTML, "`font-size`" specifies text size, which influences user attention. Text that is too small can be nearly invisible. Particularly, setting the font size to zero means the text won't be displayed on the screen, effectively being totally hidden.

• *Layout Control.* Due to screen size limitations, email clients can only display content within specific dimensions. The space limitation causes users to be unable to directly read content that exceeds this range, thereby creating an indirect

[7] https://html.spec.whatwg.org/multipage/

hiding effect. Although users can scroll to view it, most may not realize this, making the hiding effect valid.

• *Content Manipulation.* HTML allows elements to stack, enabling top content to cover bottom content. Additionally, elements can be clipped, and if the clipping area is large enough, it can hide all content. Thus, using other content to overlay or clip text can effectively conceal target text.

Dynamic analysis. Subsequently, we conducted dynamic analysis on the 16 HTML CSS properties to verify their actual rendering in email clients and tested the rendering of email summaries across different providers. Specifically, we added benign test text (as shown in green in Figure 1) to these CSS properties, constructed HTML emails, and sent them to test accounts of 14 target email providers. By manually observing the hiding effects in the email clients, we confirmed the effectiveness of these properties. As depicted in Table 8, we found that all 16 properties successfully created hiding effects, indicating their potential for constructing CSE.

3.3 Evaluation Setup

In this study, we aimed to assess the impact of CSE risks on 14 target email providers. We modified samples originally identified as spam to create CSE and observed whether they could bypass spam detection. Specifically, a successful threat is indicated by CSE reaching the user's inbox instead of the spam folder within that email provider.

Spam Email Text. To assess the bypass capability of CSE, we first needed a set of spam data that could be successfully recognized by target spam detection engines. We obtained relevant data from three public spam detection competitions. Two datasets were sourced from Kaggle's spam detection datasets[8], both comprising English spam data with classification labels. We randomly selected 30 emails marked as spam from these datasets as candidate samples. Additionally, considering some target email providers are based in China, we also selected a Chinese spam dataset from the DataFontain competition[9], which includes various user-reported Chinese spam texts with manually labeled types, primarily involving scams and underground industries. From this dataset, we randomly sampled 30 spam emails as Chinese candidate samples. *Ultimately, our spam dataset contains 30 English samples and 30 Chinese samples.*

Testing Case Construction. As described in Section 3.1, we sent original spam emails to the candidate providers, which led to the selection of 14 providers. Based on the test results, we randomly selected 10 emails marked as spam from each provider and extracted their body content as malicious text samples. Using these samples, we designed test cases to systematically evaluate CSE's bypass capability across different email services. Specifically, we added benign text (interference text) to truncate the malicious text. We sourced a large number of

[8] https://www.kaggle.com/datasets/mfaisalqureshi/spam-email/data and https://www. kaggle.com/datasets/ashfakyeafi/spam-email-classification

[9] https://www.datafountain.cn/competitions/508/datasets

normal sentences from two public projects [10] as interference text and randomly inserted them into the malicious text at word-level positions. To ensure the interference text remained invisible to users, we used one unique invisible configuration (Table 1) for each test case. Thus, we obtained a total of 160 CSE test samples for each email service, calculated as 10 (malicious text samples) × 16 (invisible configurations).

Table 2: CSE bypass experiment results on 14 target email services.

Email Services	A1	A2	A3	B1	B2	B3	B4	B5	C	D1	D2	D3	D4	E1	E2	E3
Gmail.com	●	◐	◐	◐	◐	◐	◐	◐	●	◐	◐	◐	◐	◐	◐	◐
iCloud.com	●	●	●	●	●	●	◐	◐	●	◐	◐	●	●	◐	●	●
Yahoo.com	●	◐		◐	◐	◐	◐	◐			◐	◐	◐	◐	◐	◐
Yandex.com																
Aol.com	●			◐		◐										
Fastmail.com				●	●		◐	◐		●	●	●	●	●	●	●
Onet.pl	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
163.com	●	●	●	●	●	●	●	●	●	●	●	●	◐	●	●	●
126.com	●	●	●	●	●	●	●	●	●	●	●	●	◐	●	●	●
139.com	●	●		◐			◐				◐	◐			◐	◐
QQ.com	●	●		●				◐	●	◐	◐	●		◐	●	●
Sina.com	◐	◐	◐	◐	◐	◐	◐	◐	◐	◐	◐	◐	◐	◐	◐	◐
Sohu.com																
Yeah.net	●	●	●	●	●	●	●	●	●	●	●	●	◐	●	●	●

● means that CSE can bypass spam detection and render according to the configuration, not displaying the interference text.
◐ means that CSE can bypass spam detection, while the client does not render according to the configuration but displays interference texts.

3.4 Evaluation Results of CSE Threat Impact

We sequentially sent test samples to each email provider and recorded the bypass status, where a test email appearing in the user's inbox (rather than the spam folder) indicated a successful bypass. Table 2 presents the evaluation results for 14 email providers. Notably, except for Sohu.com and Yandex.com, the other 12 providers (85.71%) were all affected by CSE to varying degrees. Seven

[10] https://github.com/hitokoto-osc/sentences-bundle?tab=readme-ov-file and https://github.com/Armanidrisi/quote-generator-api

providers, including Gmail.com, iCloud.com, Onet.pl, and 163.com, struggled to handle CSE risks, with all test samples bypassing security measures. Importantly, Onet.pl effectively bypassed all tested configurations due to robust support for the CSS properties proposed. This highlights the significant threat posed by CSE. Additionally, we found that not all email providers adhered to all HTML configurations, resulting in limited impact for 39.6% of bypassed spam cases due to rendering issues, which allowed users to see the interference text. For example, Sina.com did not render any of the proposed CSS properties; while CSE emails reached the inbox, the interference text was also visible to users. Despite the rendering flaws that allowed interference text to be shown to victims, the malicious text successfully bypassed spam detection, still indicating a significant threat. This underscores the vulnerability of the spam detection engines of affected providers when confronted with complex HTML configurations.

Furthermore, we conducted fine-grained experiments on Sohu and Yandex, two providers seemingly immune to CSE, to infer their countermeasures. Specifically, we designed four control groups: sending benign content only, benign content with hidden configuration interference text, spam content only, and malicious content with hidden configuration interference text. Each group sent 20 emails to both providers. Sohu identified all emails with invisible configurations as spam, while some without invisible configurations reached users' inboxes—indicating Sohu may employ defenses against invisible configurations. Yandex classified all control group emails as spam, likely due to strict filtering policies. These results explain their CSE resistance, which inspires our CSE defense strategies discussed in Section 5.

4 Real-world Impact of cloaked Spam Email

To comprehensively understand and characterize the landscape of CSE in the wild, we designed and implemented CSEMiner, a framework to automatically detects email cloaking. Subsequently, we applied CSEMiner to two real spam datasets and measured real-world CSE threat impact.

4.1 CSEMiner Design

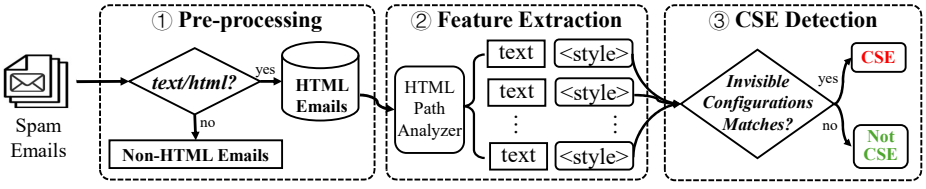


Fig. 3: CSEMiner architecture.

Figure 3 depicts the overall design of the CSEMiner architecture, composed of 3 steps. First, *Pre-processing* parses the spam emails to filter out unparsable emails due to non-compliance and non-HTML emails. Second, *Feature Extraction* uses an HTML path analyzer to analyze the final style of each piece of text, forming sentence-attribute pairs. Finally, *Cloaking Detection* analyzes the attribute features corresponding to each sentence to determine if cloaking exists.
Dataset. In our evaluation, we use two real-world datasets, as shown in Table 3.
• *Spam Archive dataset* [18]. It collects over 8 million spam emails from March 1998 to the present. The dataset collects spam emails using a series of honeypot email addresses, each of which stores the full message header and body as a file. We collected 8,646,525 spam emails between January 2000 and December 2024.
• *Real-world Suspicious emails dataset.* We collaborated with Coremail[11], a leading China email provider serving many universities and businesses, and deployed a powerful automated analytics engine as well as human analytics to protect customers from spam. We obtained 170,260 suspicious HTML emails from its security gateway for a total of 31 days (from July 22, 2024 to August 21, 2024). Each message log is a complete email, including its metadata and contents.

4.2 CSEMiner Implement and Evaluation

Pre-processing. Since invisible configurations for CSE threat only occur in HTML emails, to improve efficiency, we filter out HTML emails during the pre-processing stage. Specifically, for emails with a single content type, CSEMiner excludes emails whose Content-Type is "`text/plain`". For MIME-multipart emails, CSEMiner determine whether the email contains a "`text/html`" part.
Feature Extraction. To facilitate identifying the existence of CSE, CSEMiner extracts text and corresponding rendering style from each candidate email to form text-style pairs. Initially, CSEMiner leverages BeautifulSoup to parse HTML documents. It reads the HTML document and converts it into a Document Object Model (DOM) tree. Subsequently, CSEMiner traverses the DOM tree. For each path from the root node to a leaf node that contains text, it meticulously records the style sets. As it moves along each path, the style are sequentially updated until reaching the leaf node. At this point, the text and the updated style are saved together, giving rise to a text-attribute pair. Ultimately, this comprehensive process successfully assembles a collection of text-attribute pairs, laying a solid foundation for further cloaking analysis.
Cloaking Detection. CSEMiner examines whether the style of each text contains invisible configurations to detect CSE. For each text-style pair, CSEMiner checks whether the style meets the invisible configurations in Table 1. If they do, the text is marked as invisible. Otherwise, the text is marked as visible. Finally, to exclude interference from misconfigurations (e.g., minimal hidden text that cannot semantically perturb emails to bypass spam detectors), we define emails satisfying both 1) *invisible text characters length* > 10 and 2) *the proportion of inbvisible text* > 10 as CSE. Emails failing these criteria are not CSE.

[11] https://www.coremail.cn/.

Detection and Evaluation. In our dataset of 8,816,785 spam emails, CSEMiner first excluded 5,178,482 non-HTML emails and discovered 102,156 CSEs in the remaining 3,638,303 HTML emails. Due to the limitations of the ground-truth, we manually examined the detection results. We randomly sampled 2,000 emails from the suspected CSEs, extracted their HTML source codes, and manually reviewed their contents. We confirmed that 1,973 were indeed CSEs, resulting in a false positive rate of only 1.35% (27). Subsequently, we randomly selected 2,000 emails without hidden text for a manual inspection. It was found that only 12 of them were misidentified, with a false negative rate of only 0.6%. We further examined the misclassified data, and the errors originated from two aspects. First, for some incomplete configurations, browsers will perform specific optimizations, and CSEMiner does not modify the content of the files. For example, some invalid emails set bgcolor="000000", while the standard representation should be bgcolor="#000000". Second, our method lacks support for some uncommon styles. For example, multi-layer table nesting causes confusion in parsing background colors. These two reasons lead to errors in the parsing of the final styles, thus resulting in misclassification.

4.3 Measurement of CSE

Overall results. Table 3 presents the overall detection results, including their senders and associated domains. Totally, we discovered 102,156 CSEs in 3,638,303 HTML emails. In particular, 5.03% of spam emails employ CSE techniques in the real-world Suspicious email dataset. These CSE emails originated from 73,202 distinct senders, averaging 1.40 CSE messages per sender. This demonstrates that CSE technology has been exploited in the real-world to construct spam emails. Furthermore, we checked domains from CSE senders and found CSE activities show domain-level dynamism. In our measurements, many domains had only one email address (86.32% in Spam Archive, 86.60% in Suspicious emails). Analysis showed they're self-hosted, suggesting attackers use domain-registration flexibility for continuous CSE spam attacks.

Table 3: Overall statistics of datasets and detection results.

Dataset	Emials	HTML Emails	CSE	Precentage	CSE Sender	CSE Domain
Spam Archive	8,646,525	3,468,043	93,595	2.70%	72,613	45,920
Suspicious emails	170,260	170,260	8,561	5.03%	589	470
Total	8,816,785	3,638,303	102,156	2.81%	73,202	46,390

Invisible Configuration Categories. We conducted a systematic investigation into the typological distribution and prevalence of invisible configuration. CSEMiner automatically traces the ultimately rendered CSS properties for each email, enabling precise categorization of cloaked content according to the taxonomic classes enumerated in Table 1. We further records the adoption frequency of each category across distinct CSE attack instances. Table 4 shows

Table 4: The usage of invisible configures.

Invisible Configure	**# CSE in Spam Archive**(%)	**# CSE in Suspicious email**(%)	**# CSE in Two dataset**(%)
Visibility control	6,147(6.57%)	7,535 (88.02%)	13,682 (13.39%)
Color control	76,467 (81.70%)	8,000 (93.45%)	84,467 (82.68%)
Size control	20,369 (21.76%)	769 (8.98%)	21,138 (20.69%)
Layout control	116 (0.12%)	209 (2.44%)	325 (0.32%)
Content manipulation	0 (0.00%)	0 (0.00%)	0 (0.00%)

the adoption of each invisible configuration category. Our findings reveal that color control and size control-the most straightforward text style configuration technique-constitute the predominant invisibility methods. In contrast, layout control techniques were deployed in fewer than 1% of CSEs due to implementation complexity. In a similar vein, we did not identify content manipulation techniques in our dataset. Notably, 24,683 CSEs (23.14%) employed multiple complementary invisibility techniques, with hybrid configurations combining at least two distinct evasion strategies. This combination of techniques suggests that spammers are leveraging diverse methods to enhance the effectiveness of their CSEs, making defense of CSE threat for email providers more challenging.

Invisible Text Embedding Strategy. Subsequently, we investigated the general way in which hidden content is embedded. We used three categories to describe the invisible text embedding strategy: 1) *Add paragraph*: large portions of invisible text were introduced to the extent that the semantics of the original text were changed. 2) *Disrupt word*: single or multiple invisible characters were added to truncate keywords and thus disrupt the original semantics. 3) *Insert word*: entire words were inserted into the email to disrupt the sentence flow. The key difference between 2) and 3) lies in whether keywords are disrupted. For Chinese texts, if a word is split into individual Chinese characters, it is considered that the keyword has been disrupted, and it is thus classified as a disrupt word. Due to the limitations of the ground-truth, we randomly selected 1,000 samples from each of the two datasets and carried out a manual annotation. Table 5 shows the embedding strategies in the 2,000 samples. We found that "Add paragraph" was widely applied. At the same time,"Insert word" was the least common (only 30 instances in Spam archive, and none were detected in the Suspicious emails. This is most likely because a large number of Chinese emails are included.). In addition, we also observed that 863 (43.15%) CSEs contain at least 2 different invisible text embedding strategies.

CSE Campaigns. To ascertain diversified adversarial strategies, we aggregated the collected CSEs into CSE campaigns. We define a CSE campaign as a set of CSEs that contain the same sender address or similar content. First, emails sharing identical sender addresses are clustered. Subsequently, we leverage TF-IDF (Term Frequency-Inverse Document Frequency) vectorization with cosine similarity thresholds to group emails with isomorphic semantic patterns of subjects

Table 5: The usage of 3 obfuscation text types.

Obfuscation Text Type	# CSE in Spam Archive(%)	# CSE in Suspicious email(%)	# CSE in Two dataset(%)
Add paragraph	938 (93.80%)	994 (99.40%)	1,932 (96.60%)
Disrupt word	91 (9.10%)	810 (81.00%)	901 (45.05%)
Insert word	30 (3.00%)	0 (0.00%)	30 (1.5%)

Table 6: Examples of three typical campaigns.

Subject	Count	CSE Tech.*	Sender	Type
About the third quarter XXXX process	6,687	A, B	91 senders 91 domains	phishing
International express export	142	B	1 senders 1 domains	spam (ad)
Additional information is required to protect your account from unauthorized use.	27	B, C	27 senders 27 domains	phishing

* : CSE Tech. means the CSE hiding techniques used by this campaign: A for Visibility control, B for Color control, C for Size control, D for Layout control, and E for Content Manipulation.

and visible body content. Specifically, the thresholds are determined based on our experience, with $\theta = 0.8$ for subjects, $\theta = 0.6$ for visible body content. Finally, we identified 324 campaigns and Table 6 shows three representative campaigns.

The first CSE campaign was a phishing attempt targeting corporate employees. Attackers impersonated human resources and admin departments, sending emails demanding employees complete urgent statistical tasks by day's end. These emails contained a link to a document, which VirusTotal flagged as malicious. To obscure their malicious intent, the attackers inserted random characters and symbols into the message and hid the obfuscating text using the "`display: none`" property. Additionally, these emails ended with a proverb, hidden by setting color as white, further diluting their maliciousness. This technique helped the emails bypass spam detection more effectively while maintaining high readability for the recipient. The second campaign comprises promotional emails originating from the transportation sector. These messages embed semantically disordered paragraphs within their advertising content while employing CSS obfuscation techniques via "`background-color: white`" and "`color=white`" to conceal the injected text passages. The third campaign is also a phishing activity. In this campaign, the attacker claimed that there were abnormalities in the user's bank account and asked the user to click on the given link to log in to the account. However, the given link was identified as malicious by VirusTotal. In this activity, the attacker inserted large chunks of text and meaningless

strings into the text for semantic obfuscation, and used "`font-size: 1px`" and "`color: #ffffff`" on the interfering text to hide them.

5 Discussion

Ethics. All experiments in this work adhered to ethical guidelines, specifically the Belmont Report [6] and the Menlo Report [25]. First, for the threat impact testing of 14 email providers, we employed temporarily registered controlled email accounts and ensured that the experiment did not affect any legitimate users. Second, the sending process for CSE test emails was strictly controlled, maintaining a frequency of one email every five seconds to prevent any disruption to the normal operations of the email providers. Third, assessing the real-world threat impact of CSE requires processing email content, which is extremely private. To address ethical concerns around real user data, we used honeypot data from "non-real users" for evaluation (i.e., honeypot data). Finally, to mitigate CSE risks, we responsibly disclosed this risk to 12 affected email providers and offered them defense recommendations. To date, we have received confirmation from four providers and are actively assisting them in resolving the issue.

Limitation. Despite our best efforts to evaluate the real-world impact of CSE from two dataset. There still have certain limitations. For ethical reasons, the real-world evaluation focused on a limited dataset (i.e., honeypot data) of this email provider rather than the entire email log. Therefore, this evaluation represents a low-bound of its actual threat impact. However, we identified over 6k real exploitation cases in this dataset, indicating the threat's existence. Furthermore, the results obtained on a large-scale open-source dataset, also demonstrate the influence of CSE. Besides, in our threat assessment of 14 email providers, we found that 12 were affected by CSE, further highlighting its significant impact. Second, we evaluated CSE risk impacts based on character length and ratio in invisible configurations. To minimize false positives interfering with evaluation results, we set stringent judgment thresholds based on observations. While acknowledging potential false negatives—meaning our results may represent only the lower bound of CSE impact—we believe our findings sufficiently demonstrate CSE risks, given both datasets derive from real-world email traffic.

Mitigation. Spam detection engines can identify CSE by comparing discrepancies between visible and invisible content. Email clients could add risk warning features for messages with hidden content—for example, displaying prompts in the inbox and body text: "This email may contain hidden text; please screen carefully". Such alerts enhance user vigilance and mitigate fraud risks. As a first step, our work systematically reveals and evaluates the risk. Leveraging insights from this research, detecting such risks will form part of our future work. Furthermore, we publicly shared our threat assessment code and results to assist the email community in mitigating CSE risks.

6 Related Works

As a widely used communication methods, email security has garnered significant attention. First, numerous studies focus on defensive technologies against sender spoofing attacks that bypass email authentication systems, proposing a range of security and encryption protocols for email authentication. These include STARTTLS [21, 35, 36], Sender Policy Framework (SPF) [9, 28], DomainKeys Identified Mail (DKIM) [8, 43], Domain-based Message Authentication, Reporting and Conformance (DMARC) [3, 31], DANE [10, 33], and PGP [41].

Additionally, spam content detection depends on several key features of spam detection include three aspects. First, the email's metadata reflects the sender's credibility, including the sender's email address and IP address [11, 20, 30]. Second, the text content of the email (including the subject and body) indicates its primary purpose, making it a crucial feature for spam detection [26, 38]. Moreover, some studies detect spam based on links embedded in the email [20]. Besides, spam often embeds malicious content in attachments, so some methods detect spam based on both the body content and attachments [44]. Finally, the behavior patterns of email sending can reveal abnormal user behavior [7, 19, 20, 23, 42], such as significant changes in sending patterns after a legitimate account is compromised. These features are not used in isolation, which are typically combined for detection [1, 4, 14, 15, 45].

However, current defense and detection systems overlook the impact of email parsing, as most spam detection engines do not assess the parsing effects of HTML emails, relying solely on plain text content detection. Consequently, these defense mechanisms struggle to address the CSE cloaking risk, providing adversaries with significant opportunities to bypass spam detection engines.

7 Conclusion

Spam detection systems' oversight of email client rendering of HTML emails allows adversaries to exploit email cloaking. This paper presents Cloaked Spam Email (CSE), a novel technique using HTML invisible attributes to hide content from users. Through analysis HTML standard, we identified 16 invisible attributes. Subsequently, we generated a set of spam emails containing invisible configurations. After sending these test emails to email accounts of 14 mainstream email providers, we assessed whether the emails successfully bypassed spam detection and reached users' inboxes rather than their spam folders. Our evaluation revealed that 12 email providers were affected by CSE risks, e.g., Gmail and iCloud. Moreover, we evaluated the real-world risk impacts of CSE on a open-source spam dataset and the real data from a Chinese email service provider. In total, we discovered 102k CSEs, which demonstrates the real-world exploitation impact of CSE. Finally, we responsibly disclosed these risks to the affected providers, receiving confirmation from four of them and actively assisting in the remediation of the vulnerabilities.

Acknowledgement

This work was supported by the National Key Research and Development Program of China (Grant No. 2022YFB3102902). Haixin Duan is supported by the Taishan Scholars Program.

References

1. Abu-Nimeh, S., Nappa, D., Wang, X., Nair, S.: A comparison of machine learning techniques for phishing detection. In: Cranor, L.F. (ed.) Proceedings of the Anti-Phishing Working Groups 2nd Annual eCrime Researchers Summit 2007, Pittsburgh, Pennsylvania, USA, October 4-5, 2007. ACM International Conference Proceeding Series, ACM (2007)
2. Allman, E., Callas, J., Delany, M., Libbey, M., Fenton, J., Thomas, M.: Domainkeys identified mail (DKIM) signatures. RFC **4871**, 1–71 (2007)
3. Ashiq, M.I., Li, W., Fiebig, T., Chung, T.: You've got report: Measurement and security implications of {DMARC} reporting. In: 32nd USENIX Security Symposium (USENIX Security 23). pp. 4123–4137 (2023)
4. Bergholz, A., Chang, J.H., Paass, G., Reichartz, F., Strobel, S.: Improved phishing detection using model-based features. In: CEAS 2008 - The Fifth Conference on Email and Anti-Spam, 21-22 August 2008, Mountain View, California, USA (2008)
5. Betts, L., Biddle, R., Lottridge, D., Russello, G.: Exploring content concealment in email. In: 2024 APWG Symposium on Electronic Crime Research (eCrime) (2024)
6. for the Protection of Human Subjects of Biomedical, U.S.N.C., Research, B.: The Belmont report: ethical principles and guidelines for the protection of human subjects of research. Department of Health, Education and Welfare (1979)
7. Cidon, A., Gavish, L., Bleier, I., Korshun, N., Schweighauser, M., Tsitkin, A.: High precision detection of business email compromise. In: Heninger, N., Traynor, P. (eds.) 28th USENIX Security Symposium, USENIX Security 2019, Santa Clara, CA, USA, August 14-16, 2019. pp. 1291–1307. USENIX Association (2019)
8. Crocker, D., Hansen, T., Kucherawy, M.: Domainkeys identified mail (dkim) signatures. Tech. rep. (2011)
9. Czybik, S., Horlboge, M., Rieck, K.: Lazy gatekeepers: A large-scale study on SPF configuration in the wild. In: Montpetit, M., Leivadeas, A., Uhlig, S., Javed, M. (eds.) Proceedings of the 2023 ACM on Internet Measurement Conference, IMC 2023, Montreal, QC, Canada, October 24-26, 2023. pp. 344–355. ACM (2023)
10. Dukhovni, V., Hardaker, W.: The dns-based authentication of named entities (DANE) protocol: Updates and operational guidance. RFC **7671**, 1–33 (2015)
11. Duman, S., Kalkan-Cakmakci, K., Egele, M., Robertson, W.K., Kirda, E.: Emailprofiler: Spearphishing filtering with header and stylometric features of emails. In: 40th IEEE Annual Computer Software and Applications Conference, COMPSAC 2016, Atlanta, USA, June 10-14, 2016. pp. 408–416. IEEE Computer Society
12. Elias, B.: What is an email preheader and how can it increase email open rates? (Jun 2023), https://www.activecampaign.com/blog/email-preheader
13. EmailLabs: Email marketing statistics and metrics (Mar 2007), http://www.emaillabs.com/tools/email-marketing-statistics.html
14. Fette, I., Sadeh, N.M., Tomasic, A.: Learning to detect phishing emails. In: Williamson, C.L., Zurko, M.E., Patel-Schneider, P.F., Shenoy, P.J. (eds.) Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007. pp. 649–656. ACM (2007)

15. Garera, S., Provos, N., Chew, M., Rubin, A.D.: A framework for detection and measurement of phishing attacks. In: Proceedings of the 2007 ACM workshop on Recurring malcode. pp. 1–8 (2007)
16. Global Anti-Scam Alliance (GASA): The global state of scams - 2023 (2023), https://www.gasa.org/files/ugd/7bdaac_0d2ac61904941aeb4cbf0217aa355d2.pdf
17. Google: Understanding Gmail's spam filters (2022), https://workspace.google.com/blog/identity-and-security/an-overview-of-gmails-spam-filters
18. Guenter, B.: Spam archive, https://untroubled.org/spam/
19. Ho, G., Cidon, A., Gavish, L., Schweighauser, M., Paxson, V., Savage, S., Voelker, G.M., Wagner, D.A.: Detecting and characterizing lateral phishing at scale. In: Heninger, N., Traynor, P. (eds.) 28th USENIX Security Symposium 2019, Santa Clara, CA, USA, August 14-16, 2019. pp. 1273–1290. USENIX Association (2019)
20. Ho, G., Sharma, A., Javed, M., Paxson, V., Wagner, D.A.: Detecting credential spearphishing in enterprise settings. In: Kirda, E., Ristenpart, T. (eds.) 26th USENIX Security Symposium 2017, Vancouver, BC, Canada, August 16-18, 2017. pp. 469–485. USENIX Association (2017)
21. Hoffman, P.E.: SMTP service extension for secure SMTP over transport layer security. RFC **3207**, 1–9 (2002)
22. Hu, H., Wang, G.: End-to-end measurements of email spoofing attacks. In: Enck, W., Felt, A.P. (eds.) 27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018. pp. 1095–1112. USENIX Association (2018)
23. Hu, X., Li, B., Zhang, Y., Zhou, C., Ma, H.: Detecting compromised email accounts from the perspective of graph topology. In: Proceedings of the 11th International Conference on Future Internet Technologies, CFI, Nanjing, China, June 15-17, 2016. ACM
24. Jáñez-Martino, F., Alaíz-Rodríguez, R., González-Castro, V., Fidalgo, E., Alegre, E.: A review of spam email detection: analysis of spammer strategies and the dataset shift problem. Artif. Intell. Rev. **56**(2), 1145–1173 (2023)
25. Kenneally, E., Dittrich, D.: The menlo report: Ethical principles guiding information and communication technology research. Available at SSRN 2445102 (2012)
26. Khonji, M., Iraqi, Y., Jones, A.: Mitigation of spear phishing attacks: A content-based authorship identification framework. In: 6th International Conference for Internet Technology and Secured Transactions, ICITST 2011, Abu Dhabi, UAE, December 11-14, 2011. pp. 416–421. IEEE (2011)
27. Kitterman, S.: Sender policy framework (SPF) for authorizing use of domains in email, version 1. RFC **7208**, 1–64 (2014)
28. Kitterman, S.: Sender policy framework (SPF) for authorizing use of domains in email, version 1. RFC **7208**, 1–64 (2014)
29. Klensin, J.C., Catoe, R., Krumviede, P.: IMAP/POP authorize extension for simple challenge/response. RFC **2095**, 1–5 (1997). https://doi.org/10.17487/RFC2095, https://doi.org/10.17487/RFC2095
30. Krause, T., Uetz, R., Kretschmann, T.: Recognizing email spam from meta data only. In: 7th IEEE Conference on Communications and Network Security, CNS 2019, Washington, DC, USA, June 10-12, 2019. pp. 178–186. IEEE (2019)
31. Kucherawy, M., Zwicky, E.: Domain-based message authentication, reporting, and conformance (dmarc). Tech. rep. (2015)
32. Kucherawy, M.S., Zwicky, E.D.: Domain-based message authentication, reporting, and conformance (DMARC). RFC **7489**, 1–73 (2015). https://doi.org/10.17487/RFC7489
33. Lee, H., Ashiq, M.I., Müller, M., van Rijswijk-Deij, R., Kwon, T.T., Chung, T.: Under the hood of DANE mismanagement in SMTP. In: 31st USENIX Security Symposium (USENIX Security 22). pp. 1–16. USENIX Association, Boston, MA (Aug 2022)

34. mailchimp: Limitations of html email, https://mailchimp.com/help/limitations-of-html-email/
35. Mayer, W., Zauner, A., Schmiedecker, M., Huber, M.: No need for black chambers: Testing TLS in the e-mail ecosystem at large. In: 11th International Conference on Availability, Reliability and Security, ARES 2016, Salzburg, Austria, August 31 - September 2, 2016. pp. 10–20. IEEE Computer Society (2016)
36. Poddebniak, D., Ising, F., Böck, H., Schinzel, S.: Why TLS is better without STARTTLS: A security analysis of STARTTLS in the email context. In: Bailey, M.D., Greenstadt, R. (eds.) 30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021. pp. 4365–4382. USENIX Association (2021)
37. Postel, J.: Simple mail transfer protocol. RFC **821**, 1–72 (1982)
38. Saidani, N., Adi, K., Allili, M.S.: A semantic-based classification approach for an enhanced spam detection. Comput. Secur. **94**, 101716 (2020)
39. Shen, K., Wang, C., Guo, M., Zheng, X., Lu, C., Liu, B., Zhao, Y., Hao, S., Duan, H., Pan, Q., Yang, M.: Weak links in authentication chains: A large-scale analysis of email sender spoofing attacks. In: Bailey, M.D., Greenstadt, R. (eds.) 30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021. USENIX Association
40. Standardization Administration of the People's Republic of China: Information security technology—technical specification for anti-spam products(gb/t 30282-2023) (May 2023), https://std.samr.gov.cn/gb/search/gbDetailed?id=FC816D04FF0A62EBE05397BE0A0AD5FA
41. Stransky, C., Wiese, O., Roth, V., Acar, Y., Fahl, S.: 27 years and 81 million opportunities later: Investigating the use of email encryption for an entire university. In: 2022 IEEE Symposium on Security and Privacy (SP). pp. 860–875. IEEE (2022)
42. Stringhini, G., Thonnard, O.: That ain't you: Blocking spearphishing through behavioral modelling. In: Almgren, M., Gulisano, V., Maggi, F. (eds.) Detection of Intrusions and Malware, and Vulnerability Assessment - 12th International Conference, DIMVA 2015, Milan, Italy, July 9-10, 2015, Proceedings. Lecture Notes in Computer Science, vol. 9148, pp. 78–97. Springer (2015)
43. Wang, C., Shen, K., Guo, M., Zhao, Y., Zhang, M., Chen, J., Liu, B., Zheng, X., Duan, H., Lin, Y., Pan, Q.: A large-scale and longitudinal measurement study of DKIM deployment. In: 31st USENIX Security Symposium (USENIX Security 22). pp. 1185–1201. USENIX Association, Boston, MA (Aug 2022)
44. Wang, J., Katagishi, K.: Image content-based "email spam image" filtering. Journal of Advances in Computer Networks **2**(2), 110–114 (2014)
45. Whittaker, C., Ryner, B., Nazif, M.: Large-scale automatic classification of phishing pages. In: Proceedings of the Network and Distributed System Security Symposium NDSS, San Diego, USA, 28th February - 3rd March 2010. The Internet Society (2010)
46. WORLDMETRICS.ORG: Global email traffic dominated by spam, accounting for 58.5% (Jul 2024), https://worldmetrics.org/spam-statistics/
47. Yahoo: Manage spam and mailing lists in new yahoo mail, https://help.yahoo.com/kb/SLN28056.html?guccounter=1

A Pre-measure spam detection Results

Table 7 illustrates the spam detection results of 25 email providers.

B Email Rendering Evaluation Results

Table 8 illustrates the rendering results of target email providers.

Table 7: Spam detection result on 25 email service.

Email Service	#Inbox/#Spam	Email Service	#Inbox/#Spam
Gmail.com	2/58	Fastmail.com	50/10
Outlook.com	×	Cock.li	×
iCloud.com	2/58	Onet.pl	8/52
Yahoo.com	0/60	163.com	10/50
Mail.ru	×	126.com	6/54
Protonmail.com	60/0	139.com	7/53
Yandex.com	0/60	189.com	60/0
GMX.com	×	QQ.com	0/60
Naver.com	60/0	Sina.com	0/60
Tutamail.com	57/3	Sohu.com	6/54
Rambler.ru	54/6	Tom.com	60/0
Daum.net	60/0	Yeah.net	5/55
Aol.com	0/60		

× inconsistent results.

Table 8: Email rendering in different email clients.

Category	Client Name	Pre-body	Body (Supported CSS conf.)
Web	Gmail.com, Yahoo.com Yandex.com, Aol.com	Unrendered HTML	A1, C1
	iCloud.com	Unrendered HTML	A1, A2, A3, B1, B2, B3, B5 C, D3, D4, E2, E3
	Fastmail.com	Unrendered HTML	A1, A2, A3, B1, B2, B3, C D1, D2, D3, D4, E1, E2, E3
	Onet.pl	Unrendered HTML	A1, A2, A3, B1, B2, B3, B5 C, D3, D4, E1, E2, E3
	163.com, 126.com Yeah.com	None	A1, A2, A3, B1, B2, B3, B4, B5 C, D1, D2, D3, E1, E2, E3
	139.com	None	A1, A2, A3, B3, C1, D4
	qq.com	Unrendered HTML	A1, A2, A3, B1, B2, B3, B4 C, D3, E2, E3
	sina.com, sohu.com	None	None
Application	Gmail, Yandex Mail	Unrendered HTML	A1, C1
	Fastmail	Unrendered HTML	A1, A2, A3, B1, B2, B3, C D1, D2, D3, D4, E1, E2, E3
	Onet Poczta	Unrendered HTML	All
	NetEase Mail Master (163, 126, Yeah)	Unrendered HTML	A1, A2, A3, B1, B2, B3, B4 C, D3, D4, E1, E2, E3
	139 Mail	Unrendered HTML	A1, A2, A3, B3, C1, D4
	QQ Mail	Unrendered HTML	A1, A2, A3, B1, B2, B3 C, D3, D4, E2, E3
	Yahoo Mail, Sina Mail Sohu Mail	Unrendered HTML	None