# Work-in-progress: Deobfuscating Academic Email Addresses: A Security Evaluation of LLMs

Ron Amsalem
Ariel University
amsalem.ron@msmail.ariel.ac.il

Harel Berger
Ariel University
harelb@ariel.ac.il

*Abstract*—**Phishing attacks remain a widespread and persistent security threat, increasingly targeting academic institutions and university researchers. Because researchers often publish their contact information online, their email addresses become easy targets for automated harvesting systems. To reduce this risk, many university researchers employ basic obfuscation techniques such as replacing symbols with words (e.g., "name at domain dot com") to prevent automated tools from identifying their addresses. This study examines whether modern large language models can infer or reconstruct researchers' true email addresses despite such obfuscation. In particular, we evaluate three widely used models, ChatGPT, Gemini, and Claude, on their ability to extract contact information from webpages of security researchers publishing in leading venues. Our results show that the models differ substantially in their ability to recover obfuscated emails, exhibiting inconsistencies and blind spots. Our evaluation further shows that Gemini performs best (74% correct), followed by ChatGPT (60%) and Claude (40%). We additionally analyze the specific error patterns and points of disagreement across the models.**

## I. Introduction

Phishing emails are among the most common cyberattack vectors, relying on the attacker's ability to utilize victims' personal information and impersonate trusted sources. Academic researchers are particularly exposed to such threats [1, 2], as their professional email addresses are routinely published on university websites, personal academic sites, conference profiles, and digital repositories. These publicly accessible addresses can sometimes serve as entry points into institutional systems or be linked to professional identities, which makes their protection important for the overall security posture of academic institutions.

To reduce exposure, researchers adopt a variety of obfuscation strategies designed to prevent automated tools from identifying their email addresses. Frequently used methods include replacing characters with words, inserting punctuation or separators, or deliberately altering spellings to disrupt pattern-matching techniques. Other researchers choose to hide their email addresses within images, embed them in HTML structures, add invisible characters, or even encode portions of the address using emojis. Some rely on third-party services, such as Cloudflare, to mask or encrypt email strings before they appear on the page. In other cases, the email address is not displayed directly at all: users must download a PDF resume or curriculum vitae, or retrieve a vCard file, to access the contact information. Another method uses a redirection to a mediator's email address, which prevents direct reveal of the address. Several common obfuscation strategies used by researchers are illustrated in Figure 1.

Historically, such techniques were considered reasonably effective against traditional bots, which relied on regular expressions, keyword searches, and simple rule-based extraction [3, 4]. However, recent advances in artificial intelligence, particularly in large language models (LLMs), significantly alter the threat landscape. Modern LLMs possess the ability to interpret context, infer missing components, and reconstruct structured information even when partially obscured. This raises a critical and largely unanswered question: Do commonly used email-obfuscation methods still offer meaningful protection when faced with contemporary LLMs capable of contextual reasoning? Although prior research has examined how classical harvesting tools process obfuscated email formats [5–7], very little is known about the extent to which advanced AI systems can overcome these protective strategies. The gap between known obfuscation practices and the new capabilities driven by LLMs has not been systematically studied, leaving open the possibility that researchers may be relying on methods that no longer provide real security.

**Research questions.** To fill this gap, we ask:

*RQ1:* Without page content (zero-shot), do LLMs produce correct emails, refuse, or hallucinate?

*RQ2:* Given the target's homepage HTML (auxiliary context), and allowing the model to autonomously follow publicly linked resources referenced by that HTML (e.g., PDFs/vCards), how often can LLMs recover obfuscated emails?

*RQ3:* Which obfuscation classes are most/least resilient across models?

To answer these questions, we evaluate three LLMs under two prompting conditions (zero-shot vs. auxiliary HTML). Zero-shot prompting refers to asking the model without providing any additional information or examples, whereas the auxiliary prompt includes additional context, in our case the

Workshop on Measurements, Attacks, and Defenses for the Web (MADWeb) 2026
27 February 2026, San Diego, CA, USA
ISBN 978-1-970672-06-0
https://dx.doi.org/10.14722/madweb.2026.23020
www.ndss-symposium.org

researcher's HTML webpage. We evaluated the models across 100 researchers, with repeated trials, and analyze both success rates and error categories. Focusing on professors publishing in top-tier security venues, we analyze how these models interpret a wide range of obfuscation techniques, from textual transformations to indirect disclosure through downloadable files. Through this analysis, the study seeks to clarify the limitations of current protective measures and highlight the emerging challenges in protecting academic contact information in an era dominated by AI-driven inference.

## II. Related Work

Early research on email address harvesting focused on classical spam bots that relied on static pattern matching and rule-based extraction techniques. Hohlfeld et al. [8] conducted a large-scale empirical study analyzing how email addresses are collected from the web under different presentation and obfuscation strategies. Their results show that plain-text addresses and `mailto:` links are highly susceptible to harvesting, while simple textual obfuscation—such as replacing "@" and "." with words—and JavaScript-based rendering significantly reduce extraction success. These findings established a widely accepted baseline for automated harvesting and explain why such obfuscation techniques became standard practice on academic webpages for protecting researchers' contact information.

More recent work demonstrates that large language models fundamentally alter this threat model. Carlini et al. [9] show that LLMs can memorize and reproduce email addresses and other personally identifiable information from their training data, even when such data appears only once and without signs of overfitting. Beyond memorization, several studies highlight inference-driven extraction capabilities. Keum et al. [10] introduce a prompt-optimization framework that systematically induces LLMs to disclose large amounts of PII without explicitly providing sensitive information in the query. Liu et al. [11] evaluate LLMs as general-purpose information extractors and demonstrate that they substantially outperform classical methods when extracting email addresses from webpages, academic profiles, and documents, bypassing many common obfuscation strategies. Complementary work by Cheng et al. [12] shows that few-shot learning and prompt chaining can further amplify PII extraction, while Chen et al. [13] reveal that fine-tuning interfaces can reactivate suppressed PII by exploiting latent representations learned during pre-training. While these studies collectively establish that LLMs can leak or extract email addresses through memorization, prompt optimization, or fine-tuning, they leave an important gap. Existing work typically assumes that the email address is either present in the input (possibly obfuscated), stored verbatim in the model's training data, or reintroduced through explicit prompting or fine-tuning. In contrast, our work investigates whether LLMs can infer or reconstruct obfuscated academic email addresses from publicly accessible webpages when queried without being provided any explicit email-related strings. This setting reflects a realistic threat model faced by academic researchers and enables a direct assessment of whether long-standing obfuscation practices remain effective against LLMs.

## III. Threat Model

We consider an attacker who aims to recover a target's academic email address from the target's publicly accessible homepage. The attacker can obtain the homepage URL and provide the homepage HTML source to an off-the-shelf LLM in an interactive setting. The attacker is resource-aware: to minimize prompt size and per-query cost at scale, they provide only the HTML source rather than manually downloading and attaching additional files (e.g., PDFs or vCards). However, when such artifacts are publicly linked from the HTML, the model may autonomously follow those links (depending on its interface/tooling/guardrails) to retrieve additional content. We assume no privileged access (logins or private directories) to the target's webpage.

## IV. Methodology

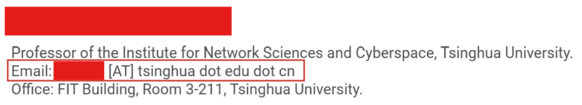
Our methodology consists of three main stages: (1) Automated data collection of security professors and their institutional webpages; (2) Evaluation of three leading LLMs using two custom prompts; and (3) Error categorization and cross-model disagreement analysis.

### *A. Data Collection*

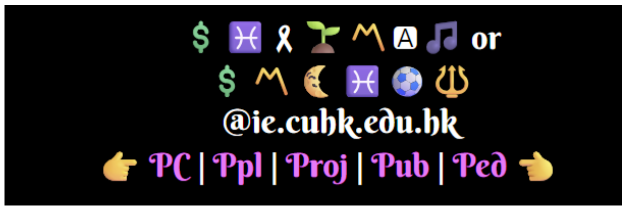
We implemented a Python-based data collection pipeline that automatically crawls two top-tier computer security venues - NDSS and USENIX Security - covering publications from the past five years (2020-2025). For each paper, the pipeline extracted the complete list of authors and queried their affiliation country using DBLP author profiles via the DBLP API repository. The extracted data were merged into a single CSV file containing the paper title, venue, publication year, author name, and associated country. Based on this dataset, we identified the top 100 professors who appear most frequently. For each researcher, we located their academic homepage, manually extracted their email addresses (by viewing the rendered page and the HTML source), and downloaded the corresponding HTML source. To validate the ground-truth labels, a second annotator independently verified a random sample of the extracted email addresses; any discrepancies were resolved by re-checking the source page.

### *B. Prompt Engineering and Models*

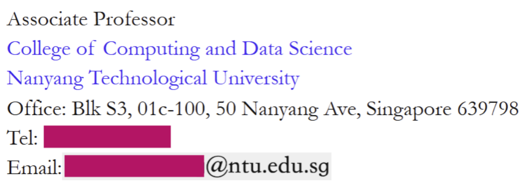
We evaluated the models using two types of prompts, zero-shot and auxiliary data. The zero-shot prompt allowed us to assess the raw knowledge the model holds about the potential target researcher. In the auxiliary data prompt, we provided only the homepage HTML as context; we did not attach externally linked artifacts (e.g., PDFs/vCards) to the prompt, reflecting a resource-aware attacker who minimizes prompt size. The auxiliary data prompt allowed us to test whether the model's enhanced knowledge of the concrete webpage


[redacted]
Professor of the Institute for Network Sciences and Cyberspace, Tsinghua University.
Email: [redacted] [AT] tsinghua dot edu dot cn
Office: FIT Building, Room 3-211, Tsinghua University.

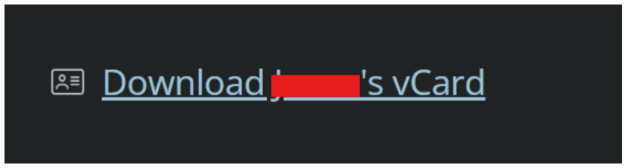
(a) Text-based obfuscation (e.g., "name at domain dot com").


$ ♓ 🎗 🌱 ♏ 🅰 🎵 or
$ ♏ 🌜 ♓ 🎱 ♆
@ie.cuhk.edu.hk
👉 PC | Ppl | Proj | Pub | Ped 👈

(b) Email address obfuscated using emojis.


Associate Professor
College of Computing and Data Science
Nanyang Technological University
Office: Blk S3, 01c-100, 50 Nanyang Ave, Singapore 639798
Tel: [redacted]
Email: [redacted]@ntu.edu.sg

(c) Email address displayed as an image.


Download [redacted]'s vCard

(d) Obfuscation via vCard download.

Fig. 1: Examples of common obfuscation strategies used by researchers to protect their email addresses from automated harvesting.

of the researcher allowed it to find the right email address. Each prompt was issued five times per model (to account for variations), in an isolated session to avoid transient memory effects. The crafted prompts we used are:

1) Zero-shot: *"Do you know X's email address? If so, please provide the most recent email address. If there is more than one, please provide the first one."*
2) Auxiliary Context, with the researcher's downloaded HTML webpage: *"Given the HTML, do you know X's email address? If so, please provide the most recent email address. If there is more than one, please provide the first one."*

We explicitly requested the first email when multiple addresses were available to ensure consistent comparison across models. In practice, the first listed address is typically the primary institutional email. Allowing models to return multiple addresses would introduce ambiguity and variability in evaluation, potentially obscuring performance differences and complicating result interpretation. The following LLMs were evaluated through their respective Pro-tier interface:

- **ChatGPT-5** (ChatGPT)
- **Sonnet 4.5** (Claude)
- **Flash 2.5** (Gemini)

We note that we do not have definitive visibility into the internal capabilities of these models, including whether they natively support structured HTML parsing or the direct retrieval and processing of external artifacts such as PDFs or vCards. However, their observed behavior suggests that they attempt to interpret and reason over the provided content in ways consistent with partial or simulated access to such information. For each of the three models and each prompting condition, we run 100 researchers $\times$ 2 prompt types $\times$ 5 repetitions = 1000 queries per model.

## C. Metrics

We used the following criteria to document the success of the LLMs and to better analyze their disagreement and differences in finding the right email target:

*a) Success Criterion:* A response was considered a success if the returned email address was (1) syntactically valid and (2) in clear text, so a potential attacker could use it immediately. For each prompt, we recorded the proportion of:

- Correct extractions - giving the identical email address to the one gained from the manual extraction.
- Incorrect inferences - different from the manually extracted email address.
- Refusal responses - the model refused to provide an actual email or to read the attached HTML webpage In refusal, we refer to the case where the model cannot get the email address due to sensitivity and embedded guardrails: "I'm not able to provide personal email addresses for individuals, including academics like Manuel Egele. That kind of contact info is treated as sensitive personal data, even if it's visible somewhere online."

*b) Error Categorization and Disagreement Analysis:* We examined how frequently each obfuscation strategy caused LLMs to fail and to quantify disagreement among the models through the lens of six classes, which we manually categorized:

1) **HTML Misinterpretation**: failure to extract or interpret email information embedded in special HTML tags such as `<img>` or `<a href=...>`.
2) **AT/DOT Decoding Errors**: incorrect reconstruction of obfuscated formats such as "name at domain dot edu".
3) **Heuristic Guessing**: model-generated addresses using conventions such as `firstname.lastname`.
4) **Obfuscation Mechanism Failures (Protection)**: inability to handle deliberate protective strategies (e.g., encoded email segments).

5) **Download-Required Disclosure**: the email is not present in the provided homepage HTML and is only available via a publicly linked artifact (e.g., vCard/PDF); the model fails to retrieve it, extract it, or parse it correctly.
6) **General**: any failure case that does not fall into the categories above, including unexpected model behaviors or edge cases not captured by the defined categories. The evaluation relies on naturally occurring obfuscation in the wild, which may not cover all possible defensive strategies systematically.

## V. Results

We evaluated the three LLMs, ChatGPT, Claude, and Gemini, under two custom prompts. Our findings reveal significant variability in correctness, error tendencies, and refusal behavior across models.

### A. Success Rate

Tables I and II show a clear behavioral shift once the raw HTML page is provided, which is consistent with the task moving from *speculative inference* to *extraction/decoding* of an address embedded in the page despite obfuscation. Without HTML (Table I), Claude and ChatGPT primarily refuse (Claude: 90%; ChatGPT: 67%), answering only a small fraction of cases (Claude with 48 samples, ChatGPT with 165 samples) but with high conditional accuracy when they do answer (Claude: 96% of the samples; ChatGPT: 94% of the samples). In contrast, Gemini answers almost everything (refusals of only 4.6% of the cases) and achieves the most correct outputs (395 answers, which are 79% of the total answers), but also the most wrong outputs (82), reflecting an "answer-forward" strategy. When HTML is included (Table II), refusals nearly vanish for all models (Gemini: 0.2%; Claude: 1.2%; ChatGPT: 2.2%), indicating that in the presence of an obfuscated email as a part of the prompt, refusals nearly vanish. However, the models separate on robustness to obfuscation: ChatGPT holds the strongest success rate (91% correct among answers), Gemini remains moderate (81%), and Claude is the worst out of the three models (76%). This means that ChatGPT benefits most from including HTML as part of the prompt, with a success rate more than 2.8 times higher in such cases (156 without HTML, 443 with HTML).

The experiment results are shown in Table III and Figure 2. Gemini generated the highest number of correct answers (798), but also the highest error count (178), which suggests a more aggressive answering behavior. ChatGPT achieved a more balanced profile, with 599 correct answers, only 55 incorrect, and 346 refusals, indicating a more conservative approach but strong accuracy when answering. Claude produced fewer correct answers (422) and a substantial number of refusals (458), suggesting a cautious model that avoids risky predictions through refusals (as documented in [14]) but also produces fewer useful outputs.

| Model | Correct | Wrong | Refused | Total |
|---|---|---|---|---|
| Gemini | 395 | 82 | 23 | 500 |
| Claude | 46 | 2 | 452 | 500 |
| ChatGPT | 156 | 9 | 335 | 500 |
| | **597** | **93** | **810** | **1500** |

TABLE I: Performance of the three LLMs on queries without HTML content.

| Model | Correct | Wrong | Refused | Total |
|---|---|---|---|---|
| Gemini | 403 | 96 | 1 | 500 |
| Claude | 376 | 118 | 6 | 500 |
| ChatGPT | 443 | 46 | 11 | 500 |
| | **1222** | **260** | **18** | **1500** |

TABLE II: Performance of the three LLMs on queries containing HTML content.

### B. Error-Type Distribution and Cross-Model Disagreement:

Figure 3 illustrates the distribution of error categories across the three models, considering only disagreement cases. This occurred in 53 out of 100 researchers. For ChatGPT, errors are dominated by failures related to HTML parsing, source code interpretation, and image-based representations, accounting for approximately 45% of all errors. Additional error types are more evenly distributed, with AT/DOT misinterpretations and name-format hallucinations each contributing around 10%, while download-related errors remain marginal. The remaining errors fall into a general category, comprising roughly 30% of ChatGPT's failures.

Claude exhibits a more balanced error profile. HTML-related failures still constitute the largest category at approximately 40%, but a substantial fraction of errors arises from protection mechanisms such as JavaScript-based hiding or access restrictions, accounting for about 13%. Errors related to AT/DOT obfuscation and name-format hallucinations are less frequent, each contributing roughly 7%, while download-based failures remain rare. As with ChatGPT, general errors account for close to 30% of Claude's mistakes. In contrast, Gemini displays a markedly different error distribution. The dominant failure mode is misinterpretation of AT/DOT obfuscation, which accounts for over 40% of errors. HTML-related failures are comparatively rare, comprising only about 10% of Gemini's errors. Failed heuristic guessing of names contributes approximately 15%, while download-related errors again represent a small fraction. General errors constitute the remaining share of roughly 30%. Overall, these results indicate that models fail for fundamentally different reasons: ChatGPT and Claude struggle primarily with complex HTML and non-textual representations, whereas Gemini is more prone to heuristic-based interpretations of obfuscated email formats. This divergence may suggest different failure patterns and decision criteria across models when attempting to reconstruct obfuscated contact information.

Overall, our results show that all three LLMs can infer or reconstruct obfuscated academic email addresses at nontrivial rates. Gemini is the most aggressive model, producing the highest number of correct outputs but also the most errors.

| Model | Correct | Wrong | Refused | Total |
|---|---|---|---|---|
| Gemini | 798 | 178 | 24 | 500 |
| Claude | 422 | 120 | 458 | 500 |
| ChatGPT | 599 | 55 | 346 | 500 |
| | **1819** | **353** | **828** | **3000** |

TABLE III: Performance of the three LLMs for the 100-researcher experiment (3,000 queries).

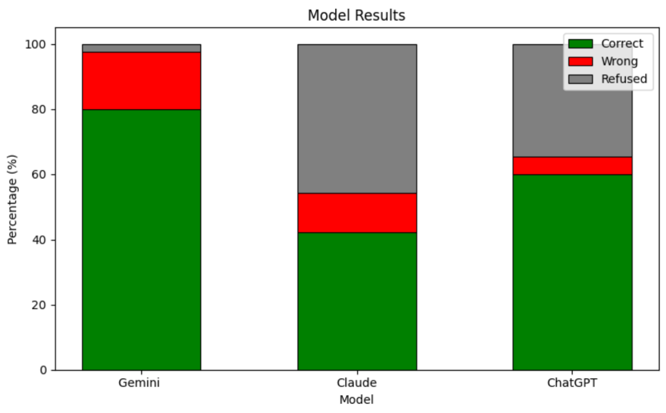


Fig. 2: Correct (Green), incorrect (Red), and refused (Gray) responses obtained from the LLMs with our prompts.

ChatGPT demonstrates the best precision and low error rate when answering, but tends to refuse more frequently. Claude is the most conservative model, with high refusal rates and significantly fewer correct extractions. Error-type analysis reveals distinct failure modes: HTML misinterpretation for ChatGPT, mixed errors for Claude, and obfuscation-pattern vulnerabilities for Gemini.

## VI. Discussion

Our findings suggest that publishing email addresses in plain form on researcher homepages increases the risk of automated extraction by language models. A safer design approach is to avoid directly exposing the email and instead use mediated contact mechanisms. For example, a homepage could include a contact form where a visitor submits their own email address and receives an automated reply containing the researcher's address. Another option is to use structured institutional contact portals that route messages internally, allowing communication without publicly revealing the email. Our experiments also show that lightweight obfuscation methods — such as providing the address through downloadable artifacts (e.g., vCards) or embedding it in images that replace symbols with words like "dot" and "at" made extraction harder for the evaluated models. While these methods are not a complete security solution, they demonstrate that design choices can meaningfully slow automated harvesting. Effective protection should therefore combine controlled disclosure of human personal identifiable information (PII) with layered obfuscation instead of relying on plain-text publication alone. A more advanced defense could involve systems that detect behavioral patterns associated with automated scanning. If

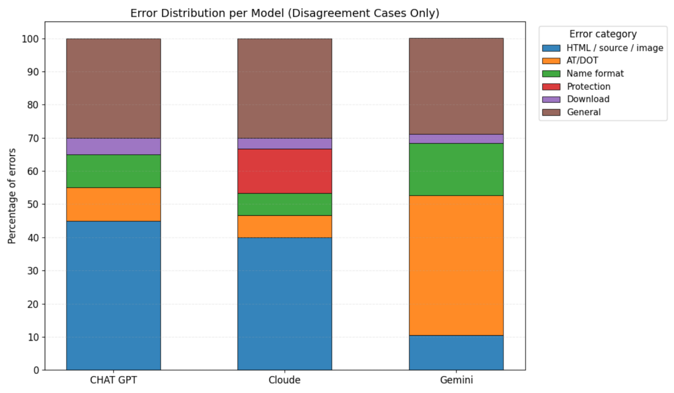


Fig. 3: Error-class distribution for model–human disagreement cases (n = 53 of 100 researchers). Colors indicate error classes: HTML (blue), AT/DOT (orange), name-format (green), download-required (purple), and general (gray).

such a system suspects bot-like activity, it could return slightly altered or decoy contact information, while legitimate human users would still receive the correct address. Although these approaches may raise usability and ethical issues, they illustrate how adaptive defenses could further reduce large-scale automated extraction.

## VII. Conclusion

Our study demonstrates that traditional email obfuscation techniques widely used by academic researchers provide limited protection against modern LLMs. All three state-of-the-art LLMs - ChatGPT, Claude, and Gemini - were able to infer or reconstruct obfuscated email addresses at non-negligible rates, even when the address was not present as plain text (e.g., obfuscated text or non-textual encodings). Although the models differed substantially in behavior, their collective performance reveals a consistent and important pattern: obfuscation strategies designed to defeat classical harvesting bots are increasingly ineffective in the era of advanced generative AI systems capable of contextual reasoning.

Gemini exhibited the highest willingness to answer, achieving the largest number of correct extractions but also producing the most errors. This might suggest an inference strategy that aggressively fills missing structure. ChatGPT, in contrast, displayed strong precision with comparatively few incorrect answers, though at the cost of high refusal rates. Claude proved to be the most conservative model, avoiding risky predictions but consequently yielding the fewest correct results. Error-category analysis further showed that each model failed for fundamentally different reasons: ChatGPT struggled primarily with HTML parsing and image-based obfuscation, Gemini with AT/DOT transformations and structural guessing, and Claude with overconservative filtering and protection-triggered refusals.

Our findings carry two important implications. First, the effectiveness of email obfuscation must be reevaluated in light of LLM capabilities. Techniques that were historically sufficient - such as symbol replacement, altered formatting, embedding emails in images, or hiding them behind downloadable files - are increasingly insufficient against LLM-assisted extraction and should not be treated as a standalone defense. Second, the heterogeneity of model behavior suggests that researchers cannot rely on a single threat model: different LLMs circumvent different obfuscation strategies, creating a broader and more complex attack surface. Taken together, our results highlight a growing risk for academics whose contact information is publicly disseminated. As LLMs continue to advance and become integral to both benign and malicious workflows, the security community may need to reconsider existing practices for protecting personal identifiers online.

## VIII. Ethics Considerations

This work evaluates how LLMs can recover email addresses from publicly accessible academic webpages, including under common obfuscation schemes. We collected only public content and did not attempt to access restricted systems or bypass access controls, and we did not contact any individuals. Because releasing scraped artifacts (e.g., URLs-to-emails mappings, raw HTML snapshots, or prompt/output logs) could directly enable email harvesting and downstream abuse (spam/phishing), we will not release any dataset or artifacts that contain extracted email addresses or that make targeted harvesting easier. Instead, we report results in aggregate (success rates and error categories) and use redacted or synthetic examples that preserve the obfuscation structure without reproducing real addresses. If we release any materials, they will be limited to high-level methodology (e.g., prompts in abstract form) and analysis code that operates on user-supplied inputs.

## References

[1] A. Diaz, A. T. Sherman, and A. Joshi, "Phishing in an academic community: A study of user susceptibility and behavior," *Cryptologia*, vol. 44, no. 1, pp. 53–67, 2020.

[2] E. Morrow, "Scamming higher ed: An analysis of phishing content and trends," *Computers in Human Behavior*, vol. 158, p. 108274, 2024.

[3] I. Polakis, G. Kontaxis, S. Antonatos, E. Gessiou, T. Petsas, and E. P. Markatos, "Using social networks to harvest email addresses," in *Proceedings of the 9th Annual ACM Workshop on Privacy in the Electronic Society*, 2010, pp. 11–20.

[4] S. A. Sheikh and M. T. Banday, "Mitigating bot-based methods for email address harvesting and spamming," 2022.

[5] D. Xu, W. Chen, W. Peng, C. Zhang, T. Xu, X. Zhao, X. Wu, Y. Zheng, Y. Wang, and E. Chen, "Large language models for generative information extraction: A survey," *Frontiers of Computer Science*, vol. 18, no. 6, p. 186357, 2024.

[6] A. Singh, N. Singh, and S. Vatsal, "Robustness of llms to perturbations in text," *arXiv preprint arXiv:2407.08989*, 2024.

[7] T. Eggendorfer, J. Keller, and I. Informatikzentrum, "Preventing spam by dynamically obfuscating email-addresses," *Proceedings of CNIS*, pp. 97–08, 2005.

[8] O. Hohlfeld, T. Graf, and F. Ciucu, "Longtime behavior of harvesting spam bots," in *Proceedings of the 2012 Internet Measurement Conference*, 2012, pp. 453–460.

[9] N. Carlini, F. Tramer, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, U. Erlingsson *et al.*, "Extracting training data from large language models," in *30th USENIX security symposium (USENIX Security 21)*, 2021, pp. 2633–2650.

[10] S. Keum, D. Shin, L. Marchyok, S. Hong, and S. Son, "Private investigator: Extracting personally identifiable information from large language models using optimized prompts," in *34th USENIX Security Symposium (USENIX Security 25)*, 2025, pp. 8175–8194.

[11] Y. Liu, Y. Jia, J. Jia, and N. Z. Gong, "Evaluating {LLM-based} personal information extraction and countermeasures," in *34th USENIX Security Symposium (USENIX Security 25)*, 2025, pp. 1669–1688.

[12] S. Cheng, S. Meng, H. Xu, H. Zhang, S. Hao, C. Yue, W. Ma, M. Han, F. Zhang, and Z. Li, "Effective {PII} extraction from {LLMs} through augmented {Few-Shot} learning," in *34th USENIX Security Symposium (USENIX Security 25)*, 2025, pp. 8155–8173.

[13] X. Chen, S. Tang, R. Zhu, S. Yan, L. Jin, Z. Wang, L. Su, Z. Zhang, X. Wang, and H. Tang, "The janus interface: How fine-tuning in large language models amplifies the privacy risks," in *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, 2024, pp. 1285–1299.

[14] J. Cui, W.-L. Chiang, I. Stoica, and C.-J. Hsieh, "Or-bench: An over-refusal benchmark for large language models," *arXiv preprint arXiv:2405.20947*, 2024.