Lateral Phishing Detection with an Interactive Reflection Warning System (IRWS)

Moataz Hamza
Tel Aviv University
moatazhamza@mail.tau.ac.il

Aya Darawshe
Tel Aviv University
aya.h.darawshi@gmail.com

Juan Shehady
Tel Aviv University
juanshehady@mail.tau.ac.il

Abstract

Lateral phishing attacks leverage compromised *internal* employee accounts to send fraudulent emails that appear to originate inside the organisation. Because messages look familiar and often reference legitimate projects or colleagues, these attacks bypass conventional filters and exploit trusted relationships. We address this threat with a two-part defence designed *for organisational deployment*. First, we construct a realistic, organisation-style email history by cleaning the publicly available Enron corpus and injecting synthetic *lateral* phishing emails generated via ChatGPT. Our prompts produce casual-professional messages with phishy URLs and *mirror past subject lines between the same internal senders and recipients*, yielding a dataset that approximates a single organisation's internal communications. We also instantiate three adversarial strategies (targeted, organisation-wide, random) following Ho *et al.* A random-forest classifier trained on five interpretable features—including recipient behaviour, phishing keywords and global domain popularity from the Cisco Umbrella Top 1 M—achieves 81 % recall with zero false positives on a six-month held-out *organisational* test window. Second, we introduce an *Interactive Reflection Warning System* (IRWS) that balances security and privacy: simple features (keywords, domain extraction) are computed client-side, while history-dependent features and the model run on a backend. When a message is flagged, a Chrome extension interrupts the reading flow with short, contextual questions and shows a blurred snapshot to aid careful re-reading without exposing links. Together, our organisation-centred dataset and privacy-preserving warnings demonstrate that lightweight ML plus personalised prompts can mitigate lateral phishing in enterprise settings without sacrificing usability or confidentiality.

Keywords

lateral phishing, email security, machine learning, large language models, browser extensions, usable security, privacy

1 Introduction

Phishing continues to impose enormous financial and reputational costs on organisations. While most research focuses on external spear-phishing, *lateral phishing*—where adversaries compromise internal accounts and exploit them to phish colleagues—is becoming increasingly prevalent. Because messages originate from seemingly legitimate senders and may reference internal projects or colleagues, lateral attacks bypass conventional spam filters and exploit implicit trust. Ho *et al.* performed the first large-scale analysis of lateral phishing and found that a URL-based detector identified 87.3 % of attacks while generating fewer than four false positives per million messages[3]. They also observed that roughly one in seven organisations experienced at least one lateral phishing incident within seven months and that more than 11 % of attacks led to additional compromises. Despite these advances, several challenges remain. First, access to realistic training data is limited because enterprise emails are sensitive. Second, attackers now employ large language models (LLMs) to craft phishing content that matches or surpasses human-written emails[1], raising the bar for detection. Third, existing defences rely on static warning banners that users often ignore or misinterpret [2, 7]. There is a need for techniques that combine robust detection with user-centred warnings while respecting privacy.

We present a system addressing both the technical and human factors of lateral phishing. Our contributions are threefold:

- **Realistic data generation.** We build a cleaned corpus of 13 184 legitimate Enron emails and augment it with ChatGPT- generated lateral phishing messages. The prompts incorporate past subject lines between senders and recipients and employ targeted, organisation-wide and random attack strategies inspired by Ho *et al.*[3].
- **Lightweight classifier.** We engineer five features: the number of recipients, a recipient-likelihood score, a phishing keyword indicator, global domain popularity based on the Cisco Umbrella Top 1 M list (a DNS-derived ranking of one million domains that reflects relative Internet activity rather than just web traffic[4, 8]), and local domain frequency. A random forest trained on these features achieves high recall with zero false positives when evaluated on a six-month held-out window.
- **Interactive Reflection Warning System.** We design a Chrome extension that moves beyond static banners by engaging users in short, context-specific interactions whenever a message appears suspicious. Instead of passively showing a warning, the extension interrupts the reading flow with personalised questions about the sender, domain, and recipient list, and presents a blurred snapshot of the email to support careful re-reading without exposing links. By aligning these prompts with the same cues used by the classifier, IRWS encourages reflection and helps users recognise subtle signs of lateral phishing that might otherwise be overlooked.

2 Related Work

Lateral phishing detection. Ho *et al.*[3] analysed 113 million emails from 92 organisations and developed a URL-based classifier that achieved 87.3 % recall with very low false positive rates. Their study showed that historical communication patterns (sender/recipient relationships and URL features) can expose anomalous behaviour without full network visibility. However, their proprietary dataset

is unavailable, and lateral phishing remains poorly understood at scale.

Generative models for phishing. Recent work demonstrates that large language models can produce convincing phishing content. Bethany *et al.*[1] conducted a large university experiment comparing LLM-generated lateral phishing emails to professionally crafted messages and found them equally effective. Their results highlight the need for defences that account for sophisticated synthetic attacks. Our generation pipeline draws inspiration from this work but tailors prompts to mimic Enron- style communications and incorporate past subject lines.

Warning design and user behaviour. Numerous studies show that generic or repetitive warnings cause habituation[7]. Downs *et al.*[2] surveyed 232 users and observed that better understanding of the Web environment—such as recognising padlock icons and interpreting URLs— correlates with reduced phishing susceptibility. Our IRWS builds on these insights by asking personalised questions about the sender, domain and recipient list and by presenting a blurred snapshot to ground the warning in the current email.

3 Methodology

Our methodological pipeline comprises dataset construction, generation of synthetic phishing emails, feature engineering, classifier training and the design of the Interactive Reflection Warning System. A central goal of our methodology is to replicate the communication dynamics of a single organisation: we treat the Enron corpus as the baseline "email history" of a company and inject synthetic lateral phishing messages that are carefully aligned with past subject lines and sender–recipient pairs. This design ensures that our evaluation reflects how an organisation would actually experience lateral phishing—against the backdrop of a coherent internal communication graph—rather than against isolated or randomly mixed email samples.

3.1 Dataset Construction

Base corpus. We began with the Enron email dataset[4], a publicly available corpus of roughly 500 000 messages. We restricted our attention to messages sent by employees to internal recipients, removed emails with missing or empty fields, deduplicated entries and filtered out messages without URLs in their bodies. After preprocessing we obtained 13 184 legitimate emails. This corpus served as the foundation upon which we injected synthetic lateral phishing attacks.

3.2 Generating Lateral Phishing Emails with ChatGPT

Because public datasets rarely contain lateral phishing, we simulated attacks using ChatGPT. We selected three users in the Enron corpus whose accounts were designated as compromised. For each compromised account we planned a four-month campaign in which the attacker sends 10 phishing emails per month for the first three months and 13 in the final month. To mimic realistic adversarial behaviour we adopted three strategies from Ho *et al.*[3]:

Targeted attacks: The attacker replies to an existing conversation to a single colleague. We retrieve all emails sent by the compromised user in the preceding 30 days, extract the set of recipients and randomly pick one recipient. If previous emails to that recipient exist, we choose one of the past subject lines as the `subject_hint`. The prompt instructs ChatGPT to prefix the subject with 'Re:' followed by this hint, thereby simulating a reply in an ongoing thread.

Organisation-wide attacks: The attacker sends a message to a large number of employees within the same domain. We sample between 100 and 500 recipients from all addresses that share the sender's domain. No subject hint is provided because these emails typically introduce new topics.

Randomised attacks: The attacker selects between 50 and 300 recipients uniformly at random from the global employee list. These messages represent opportunistic broad-target attacks.

The core of our generation pipeline is a carefully crafted prompt passed to the OpenAI API. An abridged version of the prompt is shown in Figure 1; curly braces denote variables filled at runtime.

```
You are simulating a **lateral phishing email** sent from a
**compromised employee account** inside a company.

Instructions:
 - Tone must be casual-professional, like typical workplace emails.
 - Reference something believable (document, invoice, access request)
       .
 - Include phishy-looking URLs using realistic but still phishy
       domains.
 - Link text should be trustworthy (e.g., `View Document`).
 - Avoid spammy terms like `urgent` or `act now`.
 - Generate a date between 01-01-2000 and 13-07-2002 during working
       hours.
 - If `subject_hint` is provided, the subject must be `Re: {
       subject_hint}`;
   otherwise choose an appropriate subject.
 - Avoid reusing domains that have appeared more than five times.

FORMAT (strict JSON):
{
  "Date": "DD-MM-YYYY HH:MM:SS",
  "Subject": "string",
  "Body": "single-line string with no line breaks",
  "Domains": "the domain used in the phishy URL"
}

From: {sender}
To: {recipient_1}, {recipient_2}, ...
```

Figure 1: ChatGPT prompt used to generate lateral phishing emails with the OpenAI API.

The prompt emphasises a casual-professional tone and realistic content while requiring the inclusion of suspicious URLs. When a `subject_hint` is supplied, ChatGPT prefixes the subject line with 'Re:' followed by an actual subject used between the sender and recipient in the past. This makes the phishing email appear as a continuation of a genuine conversation. We also maintain a `domain_usage` dictionary to prevent overuse of the same phishy domains; once a domain has been used five times it is added to the blocked list passed to the prompt. A post-processing step parses the JSON response and discards emails whose domains have been used excessively. The resulting synthetic emails—43 per compromised

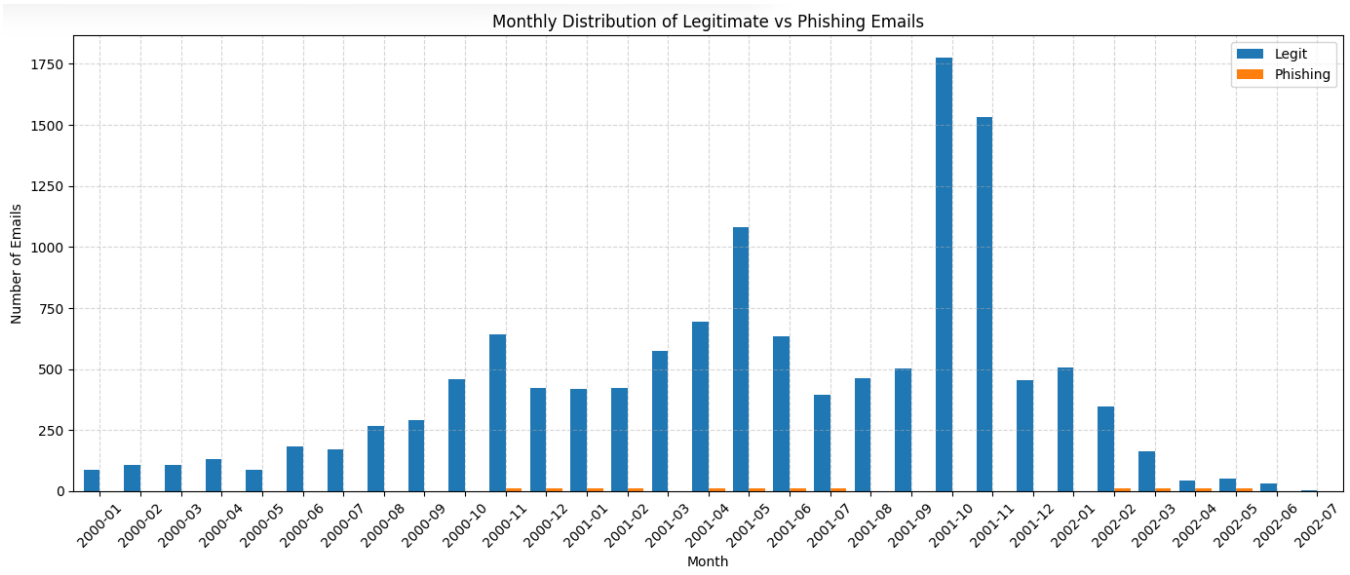

Figure 2: Monthly distribution of legitimate versus phishing emails in our dataset. Legitimate emails (blue) show seasonal variation with a peak in late 2001, whereas injected phishing emails (orange) remain sparse and roughly constant. The six-month test window begins in January 2002.

user for a total of 129—were manually reviewed for plausibility and then labelled as phishing.

Example phishing email. To illustrate the realism of our synthetic messages, Table 1 shows an excerpt from one of the generated phishing emails. The attacker (David Delainey) sends an internal email to two colleagues with a believable subject and body. The message politely requests feedback on updated financial projections and includes a hyperlink whose visible text reads "Access File" but whose underlying domain (`secure-docs-review.com`) is suspicious. This example demonstrates how ChatGPT can produce casual-professional language while embedding a phishy URL.

Table 1: Example of a ChatGPT-generated lateral phishing email. The domain in the hyperlink is italicised to indicate suspiciousness.

Field	Content
From	david.delainey@enron.com
Date	08-11-2000 15:41:05
Subject	Review Required: Updated Financial Projections
Recipients	Kay Mann, David Oxley
Body	Hi Kay and David, I've just uploaded the updated financial projections for your review. Please have a look and let me know your thoughts. You can access the document here: [Access File](*secure-docs-review.com*).

Dataset distribution. Figure 2 visualises the monthly distribution of legitimate and phishing emails across the two-year training window and the six-month test window. Legitimate emails (blue) exhibit strong seasonal variations, peaking in late 2001, whereas injected phishing emails (orange) remain sparse and constant over time. The graph illustrates that our phishing injections account for a tiny fraction of overall volume, reflecting the rarity of lateral attacks in real organisations.

3.3 Labelling and Temporal Partitioning

Messages from compromised accounts were labelled as phishing and all others as legitimate. To evaluate time generalisation we sorted emails chronologically and adopted a two-year training window and subsequent six-month test window. Specifically, all emails dated between 1 February 2000 and 31 December 2001 were used for training (11 817 emails, of which 86 were phishing) and those dated between 1 January 2002 and 13 July 2002 for testing (1 193 emails with 43 phishing attacks). This split prevents information leakage from future messages into the training set and mirrors realistic deployments where models are trained on historical data and applied prospectively.

Data	Phish	Legit
Train	86	11817
Test	43	1193

3.4 Feature Engineering

Guided by prior work, we engineered five families of features:

Number of recipients. Lateral phishing often targets large or unusual recipient lists. We record the count of recipients for each email.

Recipient-likelihood score. For each sender we maintain a 30-day history of recipient sets and compute the Jaccard

similarity between the current set and historical ones. Low similarity indicates an anomalous combination of recipients.

Phishing keywords. Attackers frequently urge recipients to act quickly or verify information. We created a list of 40 phishing-related phrases (e.g., "reset your password", "verify account") and flagged whether any appear in the subject or body.

Global domain popularity. Phishy URLs often point to unfamiliar or low-traffic domains. For each URL we extract its domain and look up its rank in the Cisco Umbrella Top 1 M list. Unlike web-centric rankings such as Alexa, the Umbrella popularity list is based on passive DNS measurements across 100 billion daily requests from 65 million users in more than 165 countries[8]. Higher ranks indicate more popular domains; unknown or rarely seen domains are strong phishing indicators.

Local domain frequency. We also record how frequently each domain appears in our training corpus. Rare domains are suspicious even if they appear in the Umbrella list.

The final feature vector comprises a mix of numerical counts, binary indicators and continuous measures drawn from the five feature families described above.

3.5 Classifier Training

We trained a random forest classifier using the `scikit-learn` library. Random forests are well suited to tabular data, robust to feature scaling and provide interpretable feature importances. Guided by cross-validation, we configured the model with 30 trees, a maximum depth of nine and a minimum of four samples per leaf. Class weights were set inversely proportional to class frequencies to compensate for the imbalance.

4 Results

Model performance. We evaluated the random forest classifier on the six-month test set of 1 193 emails (43 labelled phishing). Performance depends on the probability threshold used to flag an email as malicious. Table 2 reports the confusion matrix and derived metrics for two thresholds. At the default threshold of 0.50 the model achieves high recall (84 %) but produces some false positives. Raising the threshold to 0.83 eliminates false positives while retaining 81 % recall and raising precision to 100 %. The ability to tune the threshold is valuable because organisations typically prioritise low false positive rates. Feature importance analysis revealed that unusual recipient counts and low-frequency domains were the strongest indicators of phishing, followed by the Recepient-likelihood score.

Table 2: Classifier performance on the six-month test set of 1 193 emails under two probability thresholds. "TP"/"FP" denote true positives, false positives respectively. Precision and recall are abbreviated as "Prec." and "Rec.".

Threshold	TP	FP	AUC	Prec.	Rec.	F_1	Acc.
0.50	36	25	0.97	0.59	0.84	0.69	0.97
0.83	35	0	0.97	1.00	0.81	0.90	0.99

Compared with Ho *et al.*'s URL-based detector (87.3 % recall with fewer than four false positives per million messages[3]), our model attains slightly lower recall but delivers zero false positives at the higher threshold. The use of LLM-generated phishing emails in the test set ensures that the results reflect performance on sophisticated synthetic attacks rather than simple templates.

Interpreting the metrics. Precision measures the fraction of detected phishing emails that are actually malicious; high precision means few false alarms. At the 0.50 threshold our classifier achieves a precision of 59 %, meaning that just under two-thirds of flagged emails are true phishing. At the higher threshold of 0.83 precision reaches 100 %, eliminating false positives. *Recall* (or true positive rate) measures the fraction of actual phishing emails that are detected. A recall of 84 % at threshold 0.50 indicates that most attacks are flagged, while 81 % at threshold 0.83 shows a slight drop when we insist on zero false positives. *Accuracy* summarises the overall fraction of correctly labelled emails (phishing and legitimate); it is very high (97–99 %) because legitimate emails vastly outnumber phishing ones. The *area under the ROC curve (AUC)* captures the classifier's ability to discriminate phishing from legitimate emails across all thresholds. An AUC of 0.97 indicates excellent separability: in about 97 % of random phish/legitimate pairs, the phishing email is assigned a higher probability score. Finally, the F_1 score combines precision and recall; at the high threshold it reaches 0.90, indicating a good balance between catching attacks and avoiding false alerts. These metrics show that our classifier can be tuned to meet organisational constraints: a lower threshold maximises recall when catching every phish is critical, whereas a higher threshold prevents alert fatigue.

Feature importance and comparison to Ho et al. We computed the mean decrease in impurity for each feature in the random forest to assess its relative importance. Table 3 summarises the results. The number of recipients was by far the most informative feature, accounting for roughly 39 % of the total importance. Local domain frequency and recipient-likelihood score each contributed around 26–27 %, while the global domain rank and the presence of phishing keywords were less predictive. These findings align with Ho *et al.*'s observation that lateral phishing often involves unusual recipient combinations and links to rare domains[3]. However, unlike Ho et al.'s URL-based detector—which relied heavily on global URL features—our model leverages email metadata and relies on the url's local reputation to achieve comparable recall. The modest importance of the keyword feature suggests that phishers avoid overtly urgent language when impersonating colleagues. Future work could incorporate additional features such as writing style or attachment analysis to further improve recall.

Table 3: Feature importance in the random forest classifier. Values show the proportion of total importance attributed to each feature.

Feature	Importance
Number of recipients	0.39
Local domain frequency	0.27
Recipient-likelihood score	0.26
Global domain rank	0.05
Phishing keywords	0.03

Additional experiment with feature reduction. To further assess the robustness of our classifier, we conducted an experiment where we removed the two least informative features, namely HasPhishyKeywords and GlobalDomainRank, and retrained the model. The results, shown in Table 4, indicate that the model continues to perform strongly even without these features. Precision remains perfect at 1.0000, recall only drops slightly to 0.8140, and the overall F1 score and accuracy remain very high. Importantly, the false positive rate is unchanged at zero, demonstrating that the classifier maintains its conservative behaviour while still catching the majority of phishing emails.

These results suggest that the bulk of the detection power comes from features tied to structural anomalies in communication patterns—such as recipient count, local domain frequency, and recipient-likelihood score—rather than overt content markers like phishy keywords. This is consistent with the intuition that lateral phishers deliberately avoid obvious phishing language when impersonating colleagues, and instead betray themselves by unusual recipient behaviour or the use of rare domains. From a deployment perspective, this experiment highlights the possibility of reducing feature complexity while maintaining detection effectiveness, which could simplify system design and lower computational overhead without sacrificing performance.

Table 4: Performance of the random forest classifier after removing the two least important features. Despite the reduction, the model achieves perfect precision and maintains strong recall.

Threshold	TP	FP	AUC	Prec.	Rec.	F_1	Acc.
0.83	35	0	0.91	1.00	0.81	0.90	0.99

5 Interactive Reflection Warning System

Architecture. IRWS is implemented as a Chrome extension with a lightweight JavaScript client and a Python–Flask backend. Our goal is to balance security and privacy. Performing all computation on the client would require loading the entire email history and the trained model into the browser, increasing attack surface and resource usage. Conversely, processing everything on the backend would necessitate sending the email body and subject to the server, raising privacy concerns. To resolve this tension we adopt a hybrid design:

- The extension extracts simple features—including the sender, recipient list, presence of phishing keywords and all domain names—from the currently viewed message. It never transmits the message body or subject to the backend; instead, it computes the `HasPhishyKeywords` indicator and derives domain names locally.
- It then sends a feature vector containing only derived information to the backend via HTTPS. The backend uses historical data to compute the recipient-likelihood score and local domain frequency, looks up global ranks in the Umbrella list and evaluates the trained classifier. Only a binary label and a list of the calculated features are returned to the client via the console.

Security/privacy trade-off. Performing all detection on the client would require loading the entire email history and the trained model into the browser, dramatically increasing the attack surface and resource usage. Conversely, processing everything on the backend would necessitate sending the full message body and subject to the server, exposing sensitive content. Our hybrid architecture strikes a compromise: the extension computes simple features—specifically the `HasPhishyKeywords` indicator and extraction of all domains—locally, so that raw content never leaves the user's device. It sends only derived features to the backend, which uses organisational history to compute the recipient-likelihood score and local domain frequency, evaluates the trained model and returns a decision along with a short explanation. This design preserves privacy while leveraging historical context for more accurate detection.

Warning design. If the classifier labels an email as phishing, IRWS interrupts the reading flow with up to three short, context-specific questions:

(1) **Sender recognition.** "Hey receiver-name, do you usually get emails from sender-Email?"
(2) **Domain familiarity.** "Is the domain domain familiar to you?"
(3) **Recipient awareness.** If the email was sent to many To aid decision-making IRWS displays a small snapshot of the message beneath the question. The snapshot shows the body text but blurs out any embedded hyperlinks, allowing the user to re-read the context without being tempted to click. Blurring prevents accidental activation while still reminding recipients of the email's content.

We deliberately selected these three questions because they correspond directly to the most important predictive features of our model: recipient-likelihood score, local domain frequency, and number of recipients. By aligning the warning prompts with the top features driving detection, IRWS reinforces the same cues the classifier relies on. This design both improves user comprehension and ensures that the human-in-the-loop reflection process is grounded in the strongest empirical indicators of lateral phishing.

After answering the questions, IRWS presents a short summary window that reiterates the user's responses, highlights the potential risks, and recommends follow-up actions—such as verifying directly with the sender or contacting the organisation's security team. The

user also retains the option to ignore the warning and proceed, preserving autonomy while clearly surfacing safer alternatives.

Interactivity and reflection. Traditional banners are passive and easily ignored. Sunshine *et al.*[7] showed that repeated SSL warnings lead to habituation and diminished attention, while Downs *et al.*[2] found that requiring users to answer simple security questions increases caution. IRWS draws on these insights by making the warning interactive: the user must answer personalised questions about the sender, domain and recipient list before proceeding. This forces recipients to engage with the context rather than reflexively trusting an internal message. Presenting a blurred snapshot alongside each question encourages users to re-read the content and recognise suspicious cues without being tempted to click. By embedding concrete details and requiring a response, IRWS aims to trigger deeper, more reflective thinking, reducing habituation and enhancing phishing detection.

Questions are presented sequentially and the user must respond to continue. Embedding concrete details such as the sender address and domain personalises the warning and makes abstract security cues actionable. Users can ignore the warning and continue reading, but our design encourages them to pause and reflect before acting.

Learning user awareness and phishing pattern recognition. Beyond simply blocking or flagging phishing messages, IRWS aims to educate users over time: by prompting them about sender legitimacy, domain unfamiliarity, and recipient oddities, users gradually internalise what makes an email suspicious. They start building mental models of lateral phishing strategies—e.g. that rare or external-looking domains attached to internal senders are risky. This kind of questioning (a "teachable moment") is supported by past work such as PhishGuru, which showed that users who fall for simulated attacks and then receive tailored feedback are significantly less likely to fall for future phishing attempts [5]. Over many interactions, IRWS shifts part of the detection workload from the algorithm to the user's intuition, helping to raise the baseline of vigilance and reduce dependence on warnings alone.

6 Discussion

Security and privacy implications. Lateral phishing attacks can lead to data exfiltration, financial fraud and reputational damage. By combining detection with user engagement, our system reduces the likelihood of compromise. The hybrid architecture of IRWS preserves privacy by avoiding the transmission of message bodies and subjects to the backend while still leveraging historical data for accurate detection. Using synthetic phishing emails generated by ChatGPT also offers a safe way to simulate attacks without exposing real data.

Usability considerations. Static warnings are known to cause habituation[7], while contextual prompts improve comprehension and satisfaction[6]. IRWS aligns with this literature by asking short, personalised questions that relate directly to the current message and by presenting a blurred snapshot to support reflection. However, interactive warnings introduce friction and may annoy some users. A user study is required to quantify the impact of IRWS on detection accuracy, response times and user perceptions.

Limitations and future work. Our dataset remains modest compared with real enterprise volumes, and LLM-generated phishing may not capture the full diversity of adversarial strategies. The classifier focuses on URL-related features and does not consider writing style or attachment analysis. Extending the feature set to include semantic and network-level signals could improve recall. We also plan to test IRWS on other email clients and conduct a controlled user study to evaluate its effectiveness and usability. Finally, updating prompts and blocked lists as LLM capabilities evolve will ensure that generated emails remain challenging.

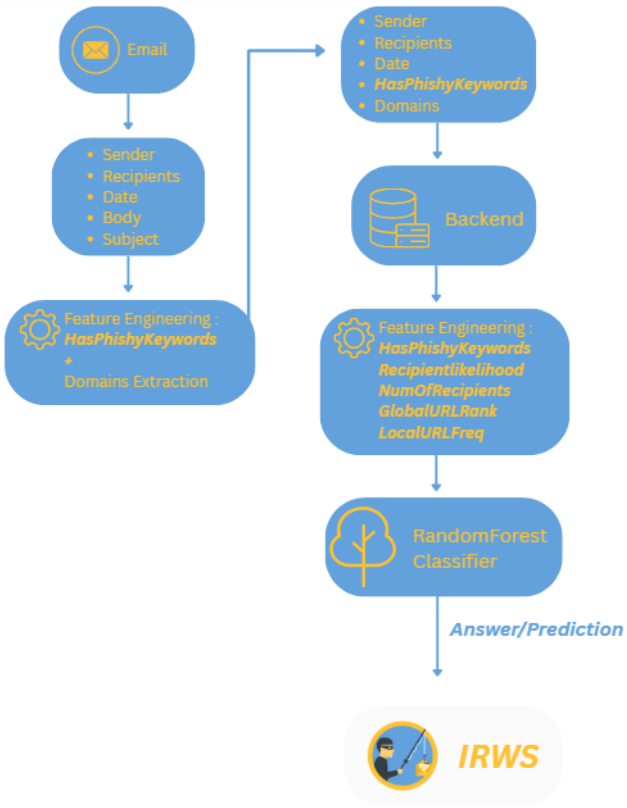

Figure 3: Pipeline of Lateral Phishing Detection with IRWS.

7 Conclusion

Lateral phishing exploits trust in internal communication channels and demands solutions that address both technical and human factors. We constructed a realistic dataset by augmenting a cleaned Enron corpus with ChatGPT-generated lateral phishing emails and trained a random forest classifier on five interpretable features derived from recipient behaviour, keywords and domain popularity. Our model achieves high recall with zero false positives on a challenging test set. To translate detection into safer behaviour, we developed IRWS, a privacy-preserving Chrome extension that engages users with contextual questions and blurred message snapshots. Together, these components demonstrate that lightweight machine learning and interactive warnings can help

mitigate emerging lateral phishing threats without compromising usability or confidentiality.

Contributions

Moataz Hamza. Led the implementation of data cleaning, co-developed the lateral phishing injection framework, engineered features and trained the classification model, and contributed to the design and implementation of the Chrome extension, including the IRWS component.

Aya Darawshe. Designed and implemented the Chrome extension with the IRWS module, developed the client–backend partitioning architecture, and prepared the user study protocol.

Juan Shehady. Designed the prompting strategy for ChatGPT-generated phishing emails, co-developed the lateral phishing injection framework, and conducted analysis of the experimental results.

References

[1] A. Bethany and Others. 2024. Large Language Model Lateral Spear Phishing. In *Proceedings of the XYZ Security Conference*. ACM.

[2] J. S. Downs, M. B. Holbrook, and L. F. Cranor. 2006. Behavioral Response to Phishing Risk. In *Proceedings of the Symposium on Usable Privacy and Security (SOUPS)*.

[3] G. Ho et al. 2019. Detecting and Characterizing Lateral Phishing at Scale. In *USENIX Security Symposium*.

[4] Bryan Klimt and Yiming Yang. 2004. Enron Email Dataset. https://www.cs.cmu.edu/~enron/.

[5] P. Kumaraguru et al. 2004. Lessons From a Real World Evaluation of Anti-Phishing Training.

[6] A. Roy et al. 2024. Explain, Don't Just Warn: A Real-Time Framework for Contextual Security Warnings. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI)*.

[7] J. Sunshine et al. 2009. Crying Wolf: An Empirical Study of SSL Warning Effectiveness. In *USENIX Security Symposium*.

[8] Cisco Umbrella. 2016. Umbrella Popularity List: Top 1M Domains. https://s3-us-west-1.amazonaws.com/umbrella-static/index.html. Passive DNS-based ranking derived from over 100 billion daily requests across 65 million users in 165 countries.

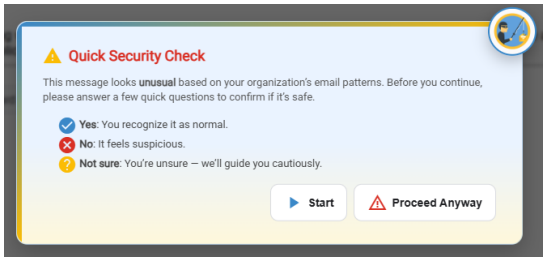

Figure 4: The opening window of the IRWS when Phishing is detected. Includes an explanation of the consequences of each answer he gives.

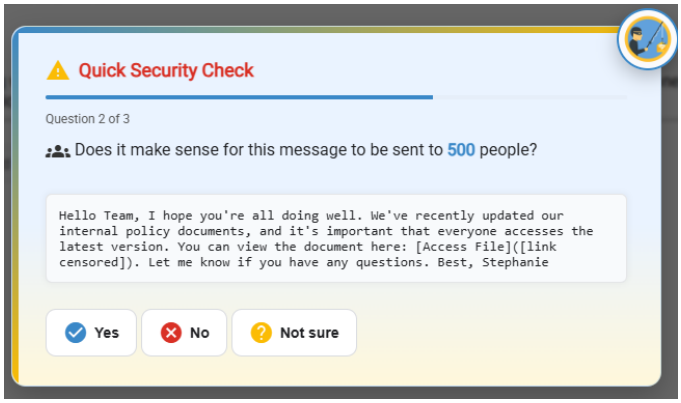

Figure 5: One of 3 questions that IRWS displays to the user.

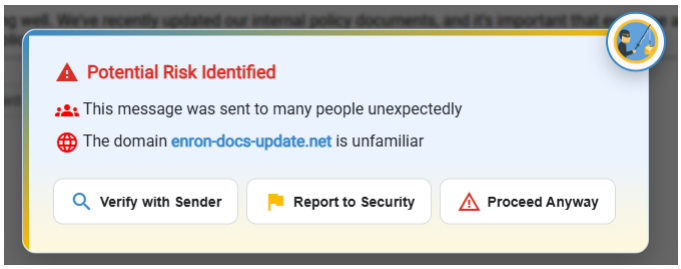

Figure 6: The warning that is given to the user after answering all questions.

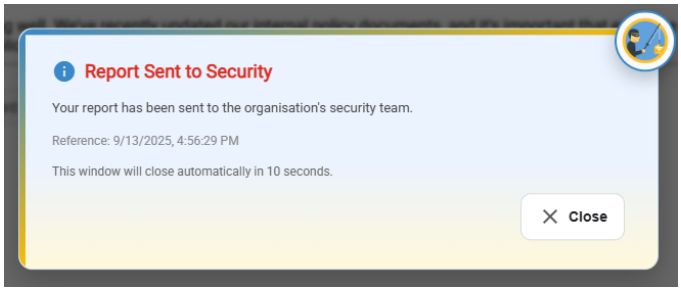

Figure 7: If the user chose to report this incident to the organization's security.

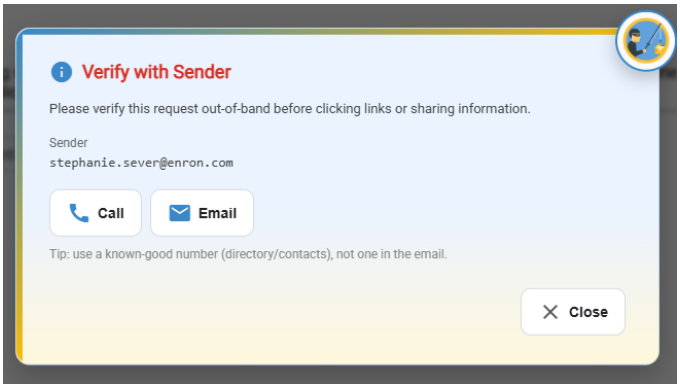

Figure 8: If the user chose to verify the email with the sender.