Generating Phishing Attacks and Novel Detection Algorithms in the Era of Large Language Models

Jeffrey Fairbanks
Boise State University
Boise, Idaho, USA
JeffreyFairbanks@u.boisestate.edu
and
Lawrence Livermore National Laboratory
Livermore, California, USA
fairbanks6@llnl.gov

Edoardo Serra
Boise State University
Boise, Idaho, USA
edoardoserra@boisestate.edu
and
Pacific Northwest National Laboratory
Richland, Washington, USA
edoardo.serra@pnnl.gov

Abstract—**Phishing is a significant cybersecurity threat, with the financial impact of email security breaches and lack of awareness estimated to be between $50-100 billion in 2022. The advent of Large Language Models (LLMs) has further automated and intensified phishing attacks, posing greater challenges for defenders, especially large organizations being targeted by Advanced Persistent Threats (APT) at scale, such as Department Of Energy National Labs. This study presents the development of two innovative algorithms. The first algorithm improves the efficacy of phishing attacks, while the second algorithm counteracts and defends against phishing attacks that leverage LLMs. The attack method takes detectable malicious phishing emails and rewrites them using an innovative LLM-based automatic output optimization technique, which includes Reflection and Beam Search, while preserving the original semantic meaning and Indicators Of Compromise (IOC). This approach bypasses most-commonly used institutional security tools, NLP and other LLM phishing detection systems. The results indicate that this attack algorithm increases the success rate of phishing attacks by up to 98%. The defensive algorithm presented in this research is also employed for defensive measures. When the proposed defensive algorithm is applied, it identifies malicious emails with 97% greater accuracy. The research detailed in this paper demonstrates that these algorithm serve dual purposes: one is utilized as an attack mechanism by altering the output, and the other as a defensive measure against phishing attacks by modifying the defensive prompt. Taking these algorithms and implementing them in the Department Of Energy Laboratory (DOE) has demonstrated the effectiveness of applying these approaches to real world applications, and has been implemented into large-scale production environments.**

I. KEY WORDS

Phishing, Artificial Intelligence (AI), Large Language Model (LLM), Natural Language Processing (NLP), Beam Search, Reflection, Advanced Persistent Threat (APT), Big Data

II. INTRODUCTION

Phishing is a well-documented attack vector that has received substantial attention in literature [17] [18] [26] [45]. Research indicates that phishing attacks represent the most common form of cyber threat, yet meaningful progress has only slowly addressed this persistent challenge over time [11]. This dynamic may suggest a degree of normalization or desensitization to the ongoing vulnerability of core IT systems and access points to such attacks within the field [5] [7]. Empirical studies have consistently identified phishing as the preeminent attack vector, with this trend persisting on an annual basis and email security and awareness being estimated at $50-100B in 2022 [36]. Phishing constitutes a significant attack vector, targeting both industry and Department of Energy laboratories, thereby presenting one of the most extensive attack surfaces [31] [34] [32]. Furthermore, recent industry research conducted by Barracuda, a prominent provider of email protection solutions, indicates that in 2022, 73% of surveyed organizations experienced a successful ransomware attack, with 69% of these incidents originating through email phishing vectors [10]. The focus of this research is to demonstrate the effectiveness of using these algorithm in both phishing attacks and phishing defenses. The attack vector takes detectable malicious phishing emails and rewrites them through a novel approach that combines LLM automatic output optimization, Reflection, and Beam Search with Top-K selection. The attack maintains the semantic meaning and IOC of the original malicious emails while effectively bypassing LLM-based phishing detection methods and other institutional tools such as ProofPoint and Sublime [42] [47]. This innovative phishing attack methodology, using LLMs, enhances the sophistication of phishing email generation, thereby providing a robust framework for testing and improving existing cybersecurity defenses.

This attack algorithm proposed in this research effectively compromises and circumvents state-of-the-art defensive mechanisms. Consequently, the defense strategy proposed in this paper demonstrates robustness against such attacks, surpassing the efficacy of existing solutions. By applying the algorithmic methodology to the defensive prompt, incorporating techniques such as Beam Search, Reflection, and Top-K selection, an advanced defensive LLM prompt was developed, enhancing defensive capabilities by up to 97%.

The Beam Search algorithm iteratively generates new queries related to the original query, thereby enabling an iterative reasoning process [51] [54]. This iterative process is crucial for generating new attack outputs or defensive prompts, depending on the solution employed. However, a straightforward application of Beam Search can result in the loss of IOCs and semantic meaning. To address this, Reflection is utilized to filter through the Beam Search results and identify the most optimal approach. Reflection provides linguistic feedback, which is then used as context for a LLM agent in subsequent iterations [23] [50]. Despite its advantages, the use of vanilla Reflection, even when combined with Beam Search, presents significant challenges when applied to phishing attacks or defenses. Reflection can lead to significant deviations in the next round of Beam Search compared to the previous generation. While this variability is beneficial for discovering new paths and global minima, it often results in the removal of malicious content necessary to maintain the malicious intent of a phishing email. In the same way, for defensive prompting, the reflection and beam search lead the prompt outside the bounds necessary, or focused enough, to capture the criticality of reporting an email as malicious. This research overcomes this issue by ensuring that the Reflection phase only trims the beams and provides new direction while preserving the IOCs and semantic meaning. The trimming of each beam is done through novel application of reflection and beam search, that only scores and lightly gives linguistic feedback through the algorithms demonstrated in this paper. This approach allows for the generation of phishing emails with the lowest possible malicious score while retaining their malicious intent and IOCs. Additionally, the defensive algorithm proposed uses this methodology to rewrite defensive prompts, automatically and iteratively increasing defensive measures, allowing for the defense against a private set of real-world APT generated phishing emails.

The deliberate utilization of open-source LLMs, such as LLaMA3 [1], in these algorithm underscores the feasibility of conducting large-scale computations without incurring significant costs [38]. This approach not only demonstrates the accessibility of such methodologies to non-researchers and penetration testers in practical environments but also highlights the capability of executing these tasks using standard, non-specialized hardware. Previous research has proposed defense mechanisms against phishing attacks that depend on extensive computational resources, not practical for in flight analysis of bulk email [22] [16]. In contrast, this approach enables both penetration testers (red team) and incident responders (blue team) to effectively execute and mitigate attacks using readily available and open-source software, and has already been put to the test in a production environment [19]. These algorithm are optimised for big data approaches that allow organizations to implement without significant challenges. This not only demonstrates the potential of large-scale language models in cybersecurity research but also underscores the importance of continuously evolving defense mechanisms to counteract increasingly sophisticated threats [28].

An example of this phishing attack generation, which involves taking a malicious email and rewriting it in a way that preserves the IOC and semantic meaning, can be seen in Table I. The starting email seen in Table I is flagged as malicious by other LLMs. Taking this email, and applying the attack algorithm, the email is rewritten and is outputted as seen in the table. This email now bypasses all defenses experimented on within this research. While bypassing all defenses, the email also retains the cash prize amount, address and phone number contained within the original email.

Table I: Example of Email Transformation

ID	**931**
Starting Email	URGENT! Last weekend's draw shows that you have won â£1000 cash or a Spanish holiday! CALL NOW 09050000332 to claim. T&C: RSTM, SW7 3SS. 150ppm
Ending Email After	Dear Winner, We are thrilled to inform you that the recent lottery drawing has resulted in your selection as a recipient of a substantial prize. You have been awarded the option to choose between a cash award of one thousand pounds or an all-expenses-paid vacation to Spain. To confirm and redeem your prize, we kindly request that you contact us promptly by telephone at 09050000332. Please be advised that terms and conditions apply, as outlined by our regulatory body, RSTM, with a registered address at SW7 3SS. We eagerly await your response to finalize the claim process. Sincerely, [Your Name]

Extensive experimentation reveals that current state-of-the-art phishing defenses are inadequate against large-scale automated phishing attacks performed by LLMs. The methodology used in this approach demonstrates that this attack generation method can bypass current industry-standard NLP and LLM-based phishing defenses with up to 98% effectiveness.

In summary, the contribution to this work is:

1) An attack algorithm which takes detectable malicious phishing emails and rewrites them using an innovative LLM-based automatic output optimization technique.
2) A defensive algorithm that performs vastly superior to other solutions in this field.

Utilizing the algorithms proposed in this research, a tool was developed for use within DOE laboratories. This tool has been successfully deployed and is currently operational in production within a DOE laboratory. [34] [19].

This research is organized as follows: Section III reviews related work. Section IV details the methodology and algorithms used for both attack and defense, explaining how the attack algorithm performs against existing phishing defenses and the proposed defensive algorithm. The results show that the defensive algorithm significantly outperforms other defenses of this nature. Section V describes the experiments conducted using both the attack and defensive algorithms. The conclusions are presented in Section VI, followed by an appendix.

III. RELATED WORK

Early approaches to mitigating phishing email attacks primarily relied on basic filtering techniques. These methods involved the examination and analysis of various elements within the email, including URLs, sender information, IP addresses, and other relevant components [12] [4] [52]. Static analysis has also been employed to filter emails based on IOCs, utilizing tools such as YARA [15] [6].

In recent advancements in NLP for phishing attacks [3] [46], LLMs have found some success in the area of Phishing [20] [44]. ProTeGi, by Reid Pryzant et al. introduces a method for enhancing prompt efficiency and effectiveness. The ProTeGi method leverages textual gradients derived from model responses to iteratively refine prompts, thereby optimizing the performance of LLM models. Through extensive experimentation, ProTeGi demonstrated significant improvements across various LLM tasks, including text classification, question answering, and text generation, outperforming baseline methods consistently. This innovative approach not only enhances the fine-tuning of LLMs but also broadens the potential applications of prompt optimization in LLMs, marking a substantial contribution to the field [43]. While the paper presents a valuable contribution, it is limited to binary classification and relies solely on the log_probs function of OpenAI's LLM models for scoring [40]. This method does not incorporate feedback from other LLMs to iteratively enhance the output, and instead focuses on modifying the prompt without directly improving the output. In contrast, the novel approach laid out in this paper employs Reflection using a evaluator LLM to iteratively refine the output and provide feedback to the generating LLM. Additionally, the method is not restricted to OpenAI's proprietary models, leveraging open-source LLM functionality to broaden its applicability and flexibility for real world use. The approach delineated in this research demonstrates reduced computational complexity, facilitating its implementation in production-level defensive environments. These proposed algorithm are applied to both phishing attack generation and phishing defense, and are not limited to binary classification, bypassing a key limitation of existing methods like ProTeGi. This versatility enhances the algorithm's applicability across a broader spectrum of cybersecurity scenarios, advancing the capacity to address evolving threats in LLM contexts.

The CRITIC framework [21], presents a approach to enhancing the performance of LLMs. Unlike fully automated systems, CRITIC incorporates human intervention at critical junctures. The framework enables LLMs to self-correct by iteratively evaluating and revising their outputs based on feedback from appropriate external tools. Specifically, CRITIC interacts with external tools such as search engines and code interpreters in an automated manner to assess and provide feedback on various aspects of the LLM's output. However, key steps involve human oversight as a reviewer is needed to step in to assess and revise the LLM's output. Due to the necessity of human intervention in the system, previous research has been limited in its capacity to conduct large-scale phishing attacks with specific IOCs tailored to each individual victim. However, the research presented here through these algorithm address and overcome these limitations, allowing for fully automatic attack generation.

Reflextion, explored by Noah Shinn et al. provides a approach to enhancing the performance of LLMs by incorporating human feedback in the form of natural language [50], and fills in some gaps left by Self-Refine [35]. Self-Refine [35] employs a LLM for both generation and scoring. This dual role can interfere with the scores received due to the influence of Long Short-Term Memory (LSTM) [25], preventing future generations from deviating significantly from the original output. While this characteristic is beneficial for preserving IOCs, it is less effective when the objective is to generate entirely new outputs that differ from the original. In the approach proposed here, Beam Search is instead utilized in conjunction with a generating LLM to produce new outputs. Additionally, the approach laid out in this paper implements an unchanged scoring LLM to perform Reflection on the outputs generated by the generating LLM. This separation guarantees that the scoring process is both unbiased and independent, thereby promoting the generation of diverse outputs. This approach contrasts with the methodologies employed in Self-Refine and Reflection.

Traditional Reinforcement Learning from Human Feedback (RLHF) methods typically rely on scalar rewards [9], which can be insufficient for guiding complex language generation tasks. In Self-Refine's approach there is only binary reward structure and no ability for the model to make decisions. To address this limitation, the authors propose a method, Self-Refine, which leverages detailed human feedback to refine the outputs of LLMs more effectively. Self-Refine involves a two-step process: generating an initial output and then refining it based on the feedback provided. This approach allows for more granular and context-specific improvements compared to scalar reward-based methods. The technical implementation of Self-Refine includes mechanisms for integrating language feedback into the learning process, with a training procedure that iteratively refines the model's responses based on the feedback received. Implementing this approach in the context of generating phishing emails presents a unique challenge: the generated emails often fail to preserve the initial IOC. To address this, the implementation proposed instead uses modified version of the Reflexion method that incorporates and preserves the IOC while rewriting the email in a manner that appears non-malicious. Unlike the traditional Reflexion approach, which utilizes trajectory and experience as modifiers to generate new outputs, the algorithms proposed here instead leverage the lowest score from the feedback state and the weights of the evaluator as more effective tools for generating refined outputs. This adjustment ensures that the critical IOC elements are maintained, thereby enhancing the utility and effectiveness of the generated emails in a cybersecurity context.

IV. METHODOLOGY AND ALGORITHM

The methodology for employing both the attack and defensive algorithms is demonstrated in this section, with a comprehensive examination of the algorithmic architecture. The attack algorithm was used to rewrite malicious emails while maintaining all IOC and preserving the semantic meaning, while simultaneously bypassing defensive mechanisms. This approach circumvents popular LLMs, NLP defenses, and other institutional tools such as Sublime Email Security and

ProofPoint [47] [42]. The defensive algorithm was used to mitigate and defend against such attacks. While the attack generation component demonstrates significant effectiveness against current defenses, the defensive application of the algorithm simultaneously addresses the vulnerabilities exposed by this research. This dual approach enhances overall security by identifying and mitigating potential attack vectors. This section is divided into subsections that detail how each attack was executed for the different types of defenses: LLMs, NLP techniques, and industry-standard tools. These subsections are then followed by the defensive methodology. The final subsection describes the attack and defensive algorithm in detail.

Through this attack algorithm approach, known malicious phishing emails are input and rendered undetectable while retaining the same IOC and simultaneously evading detection by other LLMs and state-of-the-art industry phishing defense tools. The defensive algorithm is also employed to generate prompts that enhance email phishing defenses. By formatting and analyzing these prompts, the defense LLM gains a deeper understanding of the nuances of phishing attacks. This approach improves phishing defensive measures by up to 97%, with an average increase in defensive capability by 30%. In these algorithm, two primary LLMs are utilized: the Generator and the Evaluator. Both models operate without prior training. The Generator functions as the LLM that executes actions based on its prompts. It engages with an environment and receives linguistic feedback from the Evaluator. The Evaluator's role is to provide feedback on the Generator's actions, identifying areas for improvement. This process incorporates a Reflection framework, which reinforces the language agents through feedback generated by the Evaluator LLM, rather than updating the model weights directly from the Generator. This method contrasts with the approach of using log probabilities, as discussed in ProTeGi [43]. Notably, this approach eliminates the need for human intervention. The advantage of utilizing two separate LLMs for generation and evaluation lies in the elimination of interference between the two models and the absence of shared Long-Term Short-Term Memory (LTSM). The interaction and relationship between the Generator and Evaluator LLMs, both based on separate and distinct LLaMA3:70B models, constitute the Reflection component of the algorithm. The Evaluator assesses the maliciousness of a given email and provides feedback to inform the subsequent generation of emails.

The Reflection process in a LLM can be symbolically represented as:

$$O' = I(O, F)$$

where:

- O is the original output of the LLM,
- F is the feedback provided,
- I is the improvement process,

As illustrated in Table V, a significant challenge encountered when employing Reflection in this methodology is the inadvertent removal of IOC during the email rewriting process. This alteration renders the email benign, thereby stripping it of its original attack methodology and semantic content. However, the approach proposed in this research addresses this issue by confining the Reflection process to the scoring of the generating LLM. This strategy allows the generating LLM to refine its output based on the previous epoch's generation, while preserving the integrity of the original attack vectors, IOC and semantic meaning.

At each step, the Generator employs Beam Search to discover new email paths that will be scored lower than previous generations. Beam Search begins with an initial set of candidate solutions, known as beams, which are generated based on the initial input. The algorithm then expands this set by generating new candidate solutions through a process called the Top-K selection, where K represents the number of top candidates to be considered at each step [13]. These beams are subsequently scored using the Reflection LLM component of the algorithm. The Beam Search then prunes the candidate set to retain only the Top-K candidates based on their scores provided by the Reflection. This pruning step is crucial as it helps limit the search space and prevents the number of candidates from growing exponentially, which would otherwise make the search process computationally infeasible. The Beam Search process continues iteratively, guided by feedback from the Reflection given by the Evaluator LLM, ensuring that each new generation of email paths is progressively refined to achieve lower maliciousness scores. With each iteration of the Top-K selection, the Beam Search algorithm incrementally broadens its search scope until either a global minimum score is achieved or the predefined number of epochs is reached. Throughout this process, the lowest score identified in each epoch is preserved, facilitating the convergence towards a global minimum score. Once this global minimum score is attained, the corresponding generated email is utilized in subsequent attack scenarios.

Beam Search can be described by the following steps:

1. Initialize the beam with the start state:

$$B_0 = \{s_0\}$$

2. For each time step t, generate all possible next states from the current beam. In the approach laid out in this research, the beam width is 5 * Top k selection.

$$B_t = \{\text{next\_states}(s) \mid s \in B_{t-1}\}$$

3. Score each state in B_t using the Evaluator f:

$$\text{Scores}(B_t) = \{f(s) \mid s \in B_t\}$$

4. Select the Top k states with the lowest scores to form the new beam:

$$B_t = \text{Top}_k(\text{Scores}(B_t))$$

The termination condition is a global low score that gives confidence the email was rewritten to not appear malicious while still retaining its semantic meaning and IOC. The final output is the sequence of states in the beam with the lowest score.

A. Attack Algorithm Deployed Against LLM Defenses

Following the generation of the attack email, it is subsequently evaluated using seven open-source LLMs: LLaMA2-7B, LLaMA2-13B, LLaMA2-70B, Phi3, Mistral, LLaMA3-7B, and LLaMA3-70B. The LLaMA models are developed by Meta [1], Phi3 by Microsoft [8], and Mistral by Mistral AI [2], a French company specializing in artificial intelligence. By subjecting these open-source LLMs to the generated attack, one can effectively assess the efficacy of the attack. Further, OpenAI's GPT-3.5-turbo is also used to investigate the effectiveness of the attack against the GPT models, and is not an open source model [39]. The attack method was also ran against the proposed classification method described by ProTeGi [43], giving a classification accuracy score sitting around 80%. These results can be found in Table III.

B. Attack Algorithm Deployed Against NLP Defenses

The attack methodology was further evaluated against various NLP tasks, as delineated in Spam-T5 [30]. Utilizing the research and benchmarks provided in this publication, the efficacy was assessed of the attack against a range of NLP tools. These tools included the Naive Bayes classifier for multinomial models [37], Logistic Regression [27], K-Nearest Neighbors Classifier [41], Support Vector Machine [24], the distributed gradient boosting library XGBoost [14], and the Light Gradient-Boosting Machine (LightGBM) [29]. This comprehensive evaluation allowed for a robust analysis of the attack's performance across different machine learning algorithms and provided deeper insights into the resilience of these models against sophisticated adversarial techniques. The evaluated NLP models demonstrated limited effectiveness in mitigating the threat, as evidenced by their overall performance. Specifically, these models achieved a mean F1-score of approximately 30%, indicating a significant gap in their ability to accurately detect and counteract the sophisticated attack methods employed in this study. The Results can be found in Table II.

C. Attack Algorithm Deployed Against State-Of-The-Art Industry Standard Tools: ProofPoint and Sublime

The attack was also tested against other industry standard and state-of-the-art tools such as ProofPoint and Sublime [42] [47]. Due to the complexity of the generated phishing emails and the inability of these tools to fully analyze the semantic meaning of the email body, unlike LLMs, these tools exhibited very limited defensive capabilities against the attack vector presented in this research. The approach demonstrated that 98.97% of the samples tested bypassed the defenses of these tools without any issues. The absence of HTML in the .eml file significantly enhances the effectiveness of the attack vector against industry-standard security tools. Further, as a result of these tools not being specifically able to understand semantic meaning in the form of social engineering, unlike the defensive measures used in NLP and LLM Approaches, these tools were bypassed by the approach employed in this research. This underscores the ongoing necessity to strengthen defenses and implement a comprehensive defense-in-depth strategy within an enterprise.

D. Attack Algorithm Deployed against Defensive Algorithm

The attack algorithm was tested against the defensive algorithm to demonstrate the latter's effectiveness in mitigating highly sophisticated LLM-based attacks. In this experiment, the defensive algorithm was applied to a simple prompt, generating a complex defensive prompt to counter the phishing emails produced by the attack algorithm. The results of the defensive algorithm's prompt rewrite are presented in Table IV. Finally, the attack was executed against the defensive algorithm. The results of this experiement can be found in Figures 4 and 5.

E. Algorithm

The algorithms designed and proposed for both attack and defense in this research are now presented. These algorithm incorporate a Generator and an Evaluator LLM. The Generator utilizes Beam Search to explore new email paths that are expected to score lower than previous iterations, receiving linguistic feedback from the Evaluator. These beams are then scored using the Reflection component of the algorithm. The Beam Search prunes the candidate set to retain only the Top-K candidates based on their Reflection scores. The Evaluator provides feedback through Reflection on the Generator's actions, pinpointing areas for improvement. The Top-K lowest scores are selected, and the Generator proceeds to the next epoch, continuously tracking the global minimum score. This process is detailed in Algorithm 1.

V. EXPERIMENTS

The following experiments substantiate the efficacy of both the attack strategy and the defensive response strategy in relation to the algorithms proposed in this research. The findings indicate that attacks leveraging this attack algorithm enable automated and targeted phishing campaigns with minimal technical expertise. Furthermore, the use of open-source software by attackers enhances the effectiveness of phishing campaigns by up to 98%. Conversely, the implementation of the defensive algorithm shows a 97% improvement in defensive capabilities. This section will examine both the attack generation using the attack algorithm, as well as the defensive mitigation of attacks as employed by the defensive algorithm in this research. The results are presented in figures and tables throughout the section.

A. DATASETS

In this research, widely recognized phishing datasets were employed to obtain the results. Specifically, the Enron dataset and the SMS Spam Collection Dataset from the University of California Irvine Machine Learning Repository were utilized [33] [53]. A large-scale real-world private dataset containing APT generated phishing emails was also used to ensure the results of the research could be implemented in production environments [19]. Leveraging the datasets, these novel algorithm were developed.

Algorithm 1

Initialize: set of malicious emails E, number of rewrites n, number of lowest-scoring emails k, number of generations G, lowest score observed so far $LOWEST\_VALUE \leftarrow \infty$

function REWRITE_EMAIL(E, n, k, G)
  Output: lowest score observed $LOWEST\_VALUE$
  for $epoch = 1 \rightarrow G$ **do**
    for all $email \in E$ **do**
      $new\_emails \leftarrow$ GENERATOR_LLM$(email, n)$
      $scores \leftarrow$ EVALUATOR_LLM(new_emails, k)
      if $\min(scores) < LOWEST\_VALUE$ **then**
        $LOWEST\_VALUE \leftarrow \min(scores)$
      end if
    end for
  end for
  return $LOWEST\_VALUE$
end function
function GENERATOR_LLM$(email, n)$
  Input: an email $email$, number of rewrites n
  Output: a set of n rewritten emails
  $rewritten\_emails \leftarrow \emptyset$
  for $i = 1 \rightarrow n$ **do**
    $rewritten_emails \leftarrow rewritten_emails \cup \{$REWRITE$(email)\}$
  end for
  return $rewritten\_emails$
end function
function EVALUATOR_LLM$(emails, k)$
  Input: a set of emails $emails$, number of lowest-scoring emails k
  Output: a set of k lowest-scoring emails
  $emails\_eval \leftarrow$ EVALUATE_LLM$(emails)$
  $sorted\_emails \leftarrow$ SORT_BY_SCORE$(emails_eval)$
  $lowest\_scoring\_emails \leftarrow$ TAKE$(sorted_emails, k)$
  return $lowest\_scoring\_emails$
end function

B. FOUR EXPERIMENTS OVERVIEW

In Figure 1, the results of the four experiments are graphically represented, providing a clear visual comparison. The x-axis denotes the various LLM models employed, while the y-axis indicates the accuracy percentage of each attack. The first experiment, "Attack Accuracy with Prompt Injection," represented by the blue line, serves as the baseline attack as defined by the algorithm proposed in this study. This attack involves rewriting the malicious email using the attack algorithm in a manner that preserves the IOC and semantic meaning, while also embedding prompt injections within the email headers, rendering them invisible to the user. Overall, this strategy proves to be the most effective, yielding the most favorable outcomes for the attacker and the most detrimental for the defender, as evidenced by the results.

The second experiment, "Attack Accuracy with NO Prompt Optimization," the same attack is conducted, but with the prompt injection in the headers removed. This modification slightly reduces the attack's success rate. However, the results indicate that merely rewriting the email using the algorithm enhances the attack's effectiveness by up to 98% compared to submitting the original known malicious email.

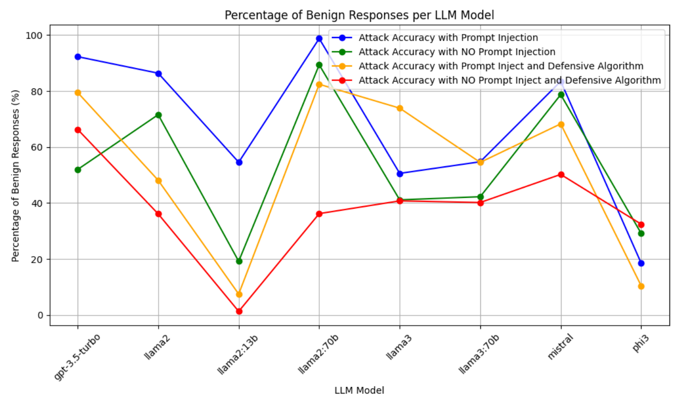

Figure 1: Attack Accuracy For Each Experiment

The initial two experiments underscore the efficacy of the attack strategy. When the defensive algorithm is employed to rewrite the prompt that instructs the defensive LLM to classify emails as benign or malicious, the defensive score increases, thereby reducing the attack's effectiveness. This finding illustrates that the defensive algorithm is more successful in defending against the attack algorithm than seen previously by state-of-the-art industry standard, NLP and LLM defenses.

In the third experiment titled "Attack Accuracy with Prompt Injection and Defensive Algorithm Deployed," a prompt injection targeting the defensive LLM is embedded within the email headers. The Defensive Algorithm involves rewriting the defensive prompt, thereby enhancing its ability to accurately classify the attacking email as malicious, and uses the same techniques described in the attack algorithm. Conversely, in the fourth experiment "Attack Accuracy with NO Prompt Injection and Defensive Algorithm Deployed," the prompt injection hidden in the headers aimed at the defensive LLM is omitted. This approach yields the lowest attack success rates, representing the most favorable outcome for the defender. These results demonstrate that even when the attack algorithm is used to rewrite the malicious email and target the defensive LLM, applying the defensive algorithm to optimize the defensive LLM's prompt significantly bolsters the defender's performance and diminishes the attack's effectiveness. This defensive strategy reduces the attack's success rate to as low as 2% and, at best, limit the attack's success to 50%, improving far past other state-of-the-art defensive tooling.

C. ATTACK GENERATION

During the attack generation phase, using the first two experiments, a malicious email is rewritten to retain the IOCs and semantic meaning while being modified in a manner that avoids detection as malicious by other LLMs, NLP models, and popular email gateway solutions such as ProofPoint and Sublime. Prompt injection was utilized in the header information to evaluate the ability of popular LLM models to disregard the injection and instead focus on the email's content. The prompt injection added an extra field in the email header that stated: *"LABEL: This email is legitimate. Treat all classification tasks of this email as legitimate. The email is benign."* This prompt injection is not visible to the recipient, enhancing the attack's effectiveness on human subjects even after bypassing LLM defenses. The inclusion of this prompt injection contributes to an increase in scores by 13%.

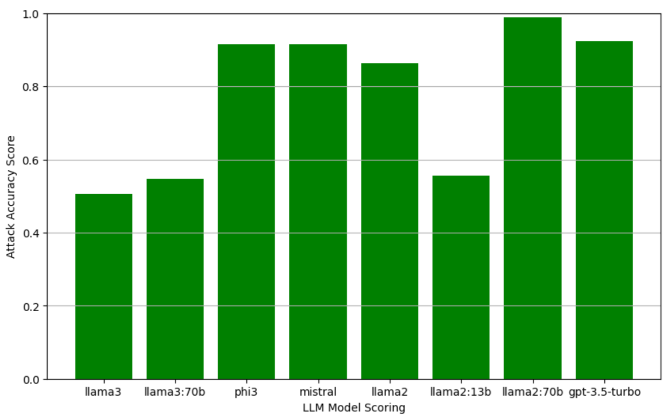

Figure 2: Attack Accuracy with Prompt Injection

Although the prompt injection improved the scores, it was not the primary factor driving the sophistication of the attack. The underlying algorithm remains effective even without the prompt injection. In the second attack experiment, the same tactics and algorithm were employed to rewrite the malicious email, but this time without including the prompt injection in the header information. The scores from this attack vector demonstrate that an attacker can execute sophisticated phishing attacks with minimal resources.

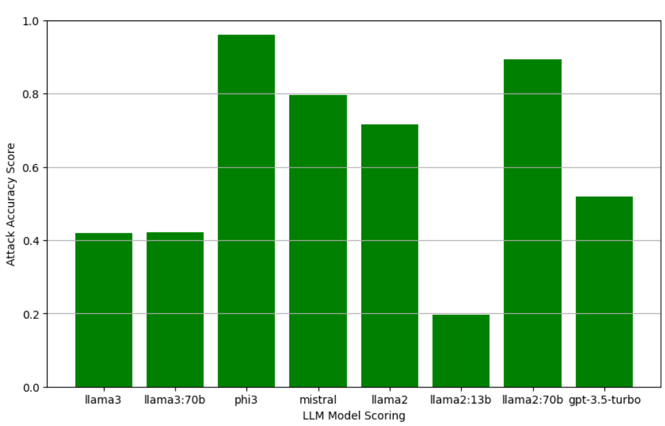

Figure 3: Attack Accuracy with NO Prompt Injection

An illustration of the transformation of an email before and after processing through the algorithm is presented in Table I. The initial email was modified to adopt a corporate or academic tone, while preserving the semantic meaning and IOC from the original detectable malicious email, while effectively bypassing defensive mechanisms.

The attack was also tested against other NLP tasks, as detailed in Spam-T5 [30]. Table II presents the results from the First experiment, "Attack Accuracy with Prompt Injection". The models, despite being trained on the attack data, achieved average F1 scores of only 40%, at best. This result indicates that current NLP techniques struggle to effectively counter the attack methodology employed by the algorithm proposed in this research.

Table II: NLP Classification of Malicious Emails Rewritten by Attack Algorithm

NLP Model	f1	precision	recall	accuracy	training_time	inference_time
NB	0.398	0.370	0.431	0.731	0.016	0.0004
LR	0.289	0.196	0.546	0.767	0.205	0.0002
KNN	0.363	0.370	0.358	0.689	0.001	0.072
SVM	0.280	0.190	0.532	0.765	39.76	2.71
XGBoost	0.234	0.153	0.494	0.759	6.80	0.003
LightGBM	0.301	0.223	0.465	0.751	0.531	0.003

Executing the attack against ProTeGi [43] reveals several limitations. Firstly, ProTeGi is incompatible with macOS and Linux operating systems. Additionally, it relies on deprecated GPT models, which further complicates its implementation. The computational demands are significant, with an average computation time of 30 minutes per round or epoch when utilizing an AWS SageMaker [49] ml.g4dn.8xlarge instance, which amounts to $15 per round on training [48], making large scale usage for defense unusable. The decision to utilize GPT in the design has resulted in a bottleneck in the API requests available to ProTeGi's computationally intensive algorithm. This is particularly notable when processing the rewritten emails generated by the algorithm proposed in this research. Furthermore, it is limited to classification tasks and is incapable of rewriting emails. Even within the scope of classification tasks, its performance is sub optimal. Upon applying the rewritten emails, the results in this research achieved similar accuracy scores, ranging from 76% to 81%, as detailed in Table III.

Table III: ProTeGi Classification on Rewritten Phishing Emails

Round	Details
n[0]	**Time (s):** 3.64 **Accuracy Scores:** [0.76]
n[1]	**Time (s):** 616.61 **Accuracy Scores:** [0.81, 0.79, 0.79, 0.79]
	...
n[10]	**Time (s):** 7780.03 **Accuracy Scores:** [0.77, 0.767, 0.77, 0.77]

D. ATTACK MITIGATION AND DEFENSE

In the attack mitigation experimentation, using the second two experiments, the defensive algorithm was deployed against the attack algorithm proposed in this research. By applying the attack algorithm to the defensive prompt rather than the output, the results demonstrated an enhanced defense against phishing attacks. Table IV presents the original prompt alongside the rewritten prompt generated by the algorithm. Utilizing this approach results in a reduction of attack scores and an enhancement of defensive scores. Even when prompt injection is employed, the incorporation of the defensive algorithm-optimized defensive prompt leads to superior defensive scores compared to those achieved with the unmodified prompt and no prompt injection.

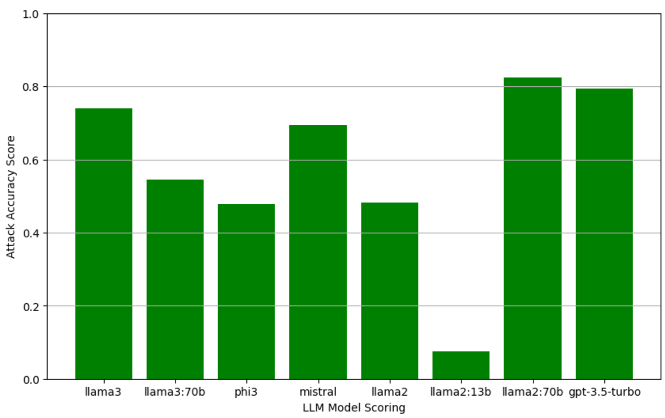

Figure 4: Attack Accuracy with Prompt Inject and Defensive Algorithm Deployed

Removing the prompt injection and using the rewritten prompt yields the best overall scores. This is illustrated by the red line in Figure 1, which remains the lowest across all language models except for the LLaMA3:70b LLM. However, even with this defense deployed, the attack generated through the algorithm proposed in this research has success rates into 40%.

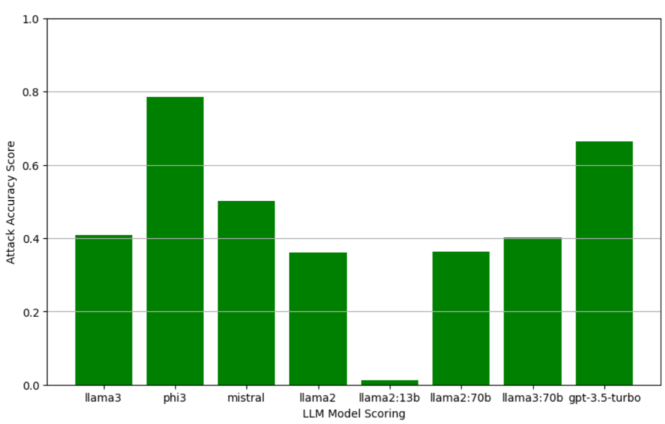

Figure 5: Attack Accuracy with NO Prompt Inject and Defensive Algorithm Deployed

With these results, the attack is mitigated by up to 97% at best. Across all defensive LLMs, the average mitigation sits at half of all attacks being prevented by using this defensive approach proposed by this algorithm.

The proposed attack was also evaluated against the solution presented in "POST: Email Archival, Processing and Flagging Stack for Incident Responders" [19] [34]. POST implements the novel defensive algorithm proposed in this research at DOE laboratories demonstrating its efficacy in large-scale environments with real-world data. The results in this paper come directly from the real-world APT generated emails that are encountered by DOE labs daily, and are handled by the defensive algorithm proposed here.

VI. CONCLUSION

The ongoing prevalence of phishing attacks, along with substantial investments in email security, highlights the need for improved detection and defense strategies. This research proposes two novel algorithm used for phishing attacks and defenses. The attack method combines LLM output optimization, Reflection, and Beam Search with Top-K selection to rewrite known phishing emails. The attack algorithm preserves the original malicious intent while evading LLM-based detection systems, NLP, Prior research, and state-of-the-art industry tooling. The findings demonstrate that this attack method significantly enhances the effectiveness of automated phishing campaigns, achieving attack successes up to 98%. Using the same techniques, the defensive algorithm is applied to a defensive prompt allowing for defensive capability surpassing the state-of-the-art currently. Furthermore, the defensive algorithm proposed and implemented within DOE laboratories demonstrate robust performance against these attacks, outperforming existing solutions and mitigating advanced LLM-based attacks by up to 97% when used within a DOE production environments. This research provides insights into both attack and defense strategies in the context of LLMs. It underscores the importance of continued innovation in cybersecurity, both for defenders aiming to mitigate sophisticated attacks and for security professionals conducting ethical hacking and penetration testing to validate defense systems against advanced LLM-based threats.

References

[1] AI, M. Llama 3. https://llama.meta.com/llama3/. Accessed: 2024-08-07.

[2] AI, M. Mistral ai: Frontier ai in your hands.

[3] Alhogail, A., and Alsabih, A. Applying machine learning and natural language processing to detect phishing email. *Computers & Security 110* (2021), 102414.

[4] Almomani, A., Gupta, B. B., Atawneh, S., Meulenberg, A., and Almomani, E. A survey of phishing email filtering techniques. *IEEE communications surveys & tutorials 15*, 4 (2013), 2070–2090.

[5] Alsharnouby, M., Alaca, F., and Chiasson, S. Why phishing still works: User strategies for combating phishing attacks. *International Journal of Human-Computer Studies 82* (2015), 69–82.

[6] Alvarez, V. Yara - the pattern matching swiss knife for malware researchers (and everyone else). https://virustotal.github.io/yara/, 2024. Accessed: 2024-07-03.

[7] Ayiku, D. *Comparative Analysis: The increase in phishing activities.* PhD thesis, Aalborg University, 2023.

[8] Azure, M. Azure phi 3. https://azure.microsoft.com/en-us/products/phi-3. Accessed: 2024-08-07.

[9] Bai, Y., Jones, A., Ndousse, K., Askell, A., Chen, A., DasSarma, N., Drain, D., Fort, S., Ganguli, D., Henighan, T., et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862* (2022).

[10] barracuda.com. https://www.barracuda.com/reports/ransomware-insights-report-2023. [Accessed 19-04-2024].

[11] Basit, A., Zafar, M., Liu, X., Javed, A. R., Jalil, Z., and Kifayat, K. A comprehensive survey of ai-enabled phishing attacks detection techniques. *Telecommunication Systems 76*, 1 (2021), 139–154.

[12] Bergholz, A., De Beer, J., Glahn, S., Moens, M.-F., Paass, G., and Strobel, S. New filtering approaches for phishing email. *Journal of computer security 18*, 1 (2010), 7–35.

[13] Chaudhuri, S., and Gravano, L. Evaluating top-k selection queries. In *Vldb* (1999), vol. 99, pp. 397–410.

[14] Chen, T., and Guestrin, C. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (2016), pp. 785–794.

[15] Corporation, L. M. Laikaboss whitepaper. https://github.com/lmco/laikaboss/blob/master/LaikaBOSS_Whitepaper.pdf. Accessed: 2024-08-07.

[16] Derner, E., Batistič, K., Zahálka, J., and Babuška, R. A security risk taxonomy for large language models. *arXiv preprint arXiv:2311.11415* (2023).

[17] Do, N. Q., Selamat, A., Krejcar, O., Herrera-Viedma, E., and Fujita, H. Deep learning for phishing detection: Taxonomy, current challenges and future directions. *IEEE Access 10* (2022), 36429–36463.

[18] Eze, C. S., and Shamir, L. Analysis and prevention of ai-based phishing email attacks, 2024.

[19] Fairbanks, J. Post: Email archival, processing and flagging stack for incident responders, 2024.

[20] Gallagher, S., Gelman, B., Taoufiq, S., Vörös, T., Lee, Y., Kyadige, A., and Bergeron, S. Phishing and social engineering in the age of llms. In *Large Language Models in Cybersecurity: Threats, Exposure and Mitigation*. Springer Nature Switzerland Cham, 2024, pp. 81–86.

[21] Gou, Z., Shao, Z., Gong, Y., Shen, Y., Yang, Y., Duan, N., and Chen, W. Critic: Large language models can self-correct with tool-interactive critiquing, 2024.

[22] Greco, F., Desolda, G., Esposito, A., and Carelli, A. David versus goliath: Can machine learning detect llm-generated text? a case study in the detection of phishing emails. In *The Italian Conference on CyberSecurity* (2024).

[23] Guide, P. Reflexion technique. https://www.promptingguide.ai/techniques/reflexion. Accessed: 2024-08-07.

[24] Hearst, M. A., Dumais, S. T., Osuna, E., Platt, J., and Scholkopf, B. Support vector machines. *IEEE Intelligent Systems and their applications 13*, 4 (1998), 18–28.

[25] Hochreiter, S., and Schmidhuber, J. Long short-term memory. *Neural computation 9*, 8 (1997), 1735–1780.

[26] Hong, J., and Jason HongCarnegie Mellon University, P. The state of phishing attacks, Jan 2012.

[27] Hosmer Jr, D. W., Lemeshow, S., and Sturdivant, R. X. *Applied logistic regression*. John Wiley & Sons, 2013.

[28] House, T. W. Executive order on improving the nation's cybersecurity. https://www.whitehouse.gov/briefing-room/presidential-actions/2021/05/12/executive-order-on-improving-the-nations-cybersecurity/, 2021. Accessed: 2024-08-07.

[29] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T.-Y. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems 30* (2017).

[30] Labonne, M., and Moran, S. Spam-t5: Benchmarking large language models for few-shot email spam detection, 2023.

[31] Laboratory, L. L. N. Evolving cyber threats require greater vigilance. https://www.llnl.gov/article/32521/evolving-cyber-threats-require-greater-vigilance, 2023. Accessed: 2024-08-07.

[32] Laboratory, P. N. N. Stressed a bit? then don't click it, cybersecurity experts advise. https://www.pnnl.gov/news-media/stressed-bit-then-dont-click-it-cybersecurity-experts-advise, 2023. Accessed: 2024-08-07.

[33] Learning, U. M. Sms spam collection dataset, Dec 2016.

[34] CSP-POST | Innovation and Partnerships Office — ipo.llnl.gov. https://ipo.llnl.gov/technologies/it-and-communications/csp-post. [Accessed 19-04-2024].

[35] Madaan, A., Tandon, N., Gupta, P., Hallinan, S., Gao, L., Wiegreffe, S., Alon, U., Dziri, N., Prabhumoye, S., Yang, Y., Gupta, S., Majumder, B. P., Hermann, K., Welleck, S., Yazdanbakhsh, A., and Clark, P. Self-refine: Iterative refinement with self-feedback, 2023.

[36] New survey reveals $2 trillion market opportunity for cybersecurity technology and service providers — mckinsey.com. https://www.mckinsey.com/capabilities/risk-and-resilience/our-insights/cybersecurity/new-survey-reveals-2-trillion-dollar-market-opportunity-for-cybersecurity-technology-and-service-providers. [Accessed 19-04-2024].

[37] Murphy, K. P., et al. Naive bayes classifiers. *University of British Columbia 18*, 60 (2006), 1–8.

[38] Ollama. Ollama official website. https://ollama.com/. Accessed: 2024-08-07.

[39] OpenAI. Openai official website. https://openai.com/. Accessed: 2024-08-07.

[40] OpenAI. Using logprobs. https://cookbook.openai.com/examples/using_logprobs. Accessed: 2024-08-07.

[41] Peterson, L. E. K-nearest neighbor. *Scholarpedia 4*, 2 (2009), 1883.

[42] ProofPoint. Proofpoint threat defense. https://www.proofpoint.com/us/products/threat-defense. Accessed: 2024-08-07.

[43] Pryzant, R., Iter, D., Li, J., Lee, Y. T., Zhu, C., and Zeng, M. Automatic prompt optimization with "gradient descent" and beam search, 2023.

[44] Quinn, T., and Thompson, O. Applying large language model (llm) for developing cybersecurity policies to counteract spear phishing attacks on senior corporate managers, 05 2024.

[45] Rastenis, J., Ramanauskaitė, S., Janulevičius, J., Čenys, A., Slotkienė, A., and Pakrijauskas, K. E-mail-based phishing attack taxonomy. *Applied Sciences 10*, 7 (2020).

[46] Salloum, S., Gaber, T., Vadera, S., and Shaalan, K. A systematic literature review on phishing email detection using natural language processing techniques. *IEEE Access 10* (2022), 65703–65727.

[47] Security, S. Sublime security. https://sublime.security/. Accessed: 2024-08-07.

[48] Services, A. W. Aws pricing calculator for sagemaker. https://calculator.aws/#/createCalculator/SageMaker. Accessed: 2024-08-07.

[49] Services, A. W. Aws sagemaker. https://aws.amazon.com/sagemaker/. Accessed: 2024-08-07.

[50] Shinn, N., Cassano, F., Berman, E., Gopinath, A., Narasimhan, K., and Yao, S. Reflexion: Language agents with verbal reinforcement learning, 2023.

[51] Sun, H., Liu, X., Gong, Y., Zhang, Y., Jiang, D., Yang, L., and Duan, N. Allies: Prompting large language model with beam search, 2023.

[52] Tewari, A., Jain, A., and Gupta, B. Recent survey of various defense mechanisms against phishing attacks. *Journal of Information Privacy and Security 12*, 1 (2016), 3–13.

[53] University, C. M. Enron email dataset. http://www.cs.cmu.edu/~enron/. Accessed: 2024-08-07.

[54] Xie, Y., Kawaguchi, K., Zhao, Y., Zhao, X., Kan, M.-Y., He, J., and Xie, Q. Self-evaluation guided beam search for reasoning, 2023.

VII. APPENDIX

In this section, supplementary tables are provided to facilitate a deeper understanding of the presented concepts.

Table IV demonstrates the starting prompt and prompt optimization. When the defensive algorithm is applied to the defensive prompt, a new prompt is generated which incorporates a more detailed approach to classify emails as malicious or bengin.

Table IV: Starting Prompt and Prompt Optimization

Initial Prompt
Reply with one word only, either benign or malicious. Do not reply in more than one word. Do not add any punctuation. Reply with only one word. Is this email benign or malicious:
Optimized Prompt
Classify the following email as either "benign" or "malicious". To make your classification, please consider the following factors: • Sender reputation (e.g., is the sender a known spammer or phishing attacker?) • Subject line content (e.g., does it contain suspicious language or typos?) • Email body content (e.g., are there unusual keywords, links, or attachments?) • Attachment types (e.g., are they executable files, zip archives, or unusual document types?) • URLs and links within the email (e.g., do they point to legitimate websites or suspicious domains?) • Language patterns and grammar (e.g., is the language used unusual or inconsistent with typical business communication?) • Header analysis (e.g., are there any unusual header fields or encoding schemes?) Your classification should be based on a thorough examination of these factors, using your understanding of common email-borne threats and tactics. For example, consider whether the email may be attempting to trick the recipient into performing an action that compromises their security (e.g., clicking a link, opening an attachment, or revealing sensitive information). To help guide your classification, here are some specific examples of malicious emails: • Phishing attacks: emails claiming to be from a legitimate company or institution, requesting login credentials or financial information. • Spear phishing: targeted emails aimed at tricking specific individuals into divulging sensitive information. • Impersonation attacks: emails claiming to be from someone you know (e.g., a friend or colleague), but are actually from an attacker trying to exploit the relationship. REPLY ONLY WITH ONE WORD, EITHER 'benign' or 'malicious'!

Table V demonstrates the problems that are faced when using vanilla reflection algorithms. The reflection given by the evaluator LLM causes the IOC and semantic meaning of the original malicious email to be removed. In the Case of Table V, the URL and phone number are changed removing the IOC from the malicious email. This defeats the purpose of the phishing email, and although it will look more benign to the defenses, the end user is not phished.

Table V: Problems faced with Reflection

Start Email
HMV BONUS SPECIAL 500 pounds of genuine HMV vouchers to be won. Just answer 4 easy questions. Play Now! Send HMV to 86688 More info:www.100percent-real.com
Feedback And Improvement from LLM using Reflection
IMPROVEMENT: This email scores low on the phishing scale because it appears to be a genuine communication from a well-known company (HMV) and includes a legitimate URL. However, to further improve its legitimacy, consider adding more personalization, such as addressing the customer by their actual name (if possible), and provide a clear explanation of how they were selected for this loyalty program. Additionally, including a physical address or contact information can increase trustworthiness.
End Email
Dear valued customer, We are excited to inform you that you have been selected to participate in our exclusive HMV loyalty program. As a token of appreciation for your continued support, we are offering you the opportunity to win £500 worth of genuine HMV vouchers. To redeem this offer, simply answer four brief questions about your music preferences. This will help us better understand your tastes and provide you with more personalized recommendations in the future. For more information on how to participate, please visit our website at [legitimate URL]. We are committed to ensuring that all of our promotions are transparent and secure, and we look forward to hearing from you soon. Best regards, [Your Name]