International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 04 | April - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

Capture-the-Phish

(**Prof.Pankaj Chandre**, Assistant Professor in Department of **Computer Scirnce and Engineering, MIT School of Computing**.)

Authors Name/s
Kanak Lingwat
Department of
Computer Science and Engineering
MIT ADT University
Pune-412201,India
kanaklingwat2904@gmail.com

Authors Name/s
Prachi Katrodia
Department of
Computer Science and Engineering
MIT ADT University
Pune-412201,India
patelprachi991@gmail.com

Authors Name/s
Tejas Vidhate
Department of
Computer Science and Engineering
MIT ADT University
Pune-412201,India
tejasvidhate2016@gmail.com

Authors Name/s
Swethanshi Patro
Department of
Computer Science and Engineering
MIT ADT University
Pune-412201,India
patroswethanshi@gmail.com

Abstract*—**Phishing attacks remain one of the most prevalent and dangerous forms of social engineering in cybersecurity. They exploit human error to bypass technological defenses, compromising credentials and sensitive data. This paper presents ***Capture the Phish, an integrated AI-powered framework that combines transformer-based natural language processing, ***explainable AI (XAI),*** and ***privacy-preserving federated learning***. The system transforms phishing detection into an adaptive, interactive learning environment that simultaneously enhances machine intelligence and human awareness. Through ***deep semantic modeling, contextual explanations, and decentralized model training***, the framework provides a scalable, privacy-conscious, and explainable phishing defense system.

Keywords—Phishing Detection, Explainable AI, Federated Learning, Cybersecurity, BERT, RoBERTa, Data Privacy.

I. INTRODUCTION

Phishing continues to be one of the most persistent and damaging cyber threats, serving as a primary vector for credential theft, identity compromise, and large-scale data breaches. Despite extensive security awareness programs, attackers continually evolve their tactics to bypass traditional defenses. Reports indicate that phishing remains responsible for a major share of security incidents across both organizations and individuals worldwide.

Traditional filtering methods, which rely heavily on static rules, keyword lists, or signature-based detection, often fail to identify novel or cleverly disguised phishing attacks. These methods lack the contextual understanding needed to interpret human language nuances, making them ineffective against modern, socially engineered threats. As a result, organizations struggle to keep pace with emerging attack patterns and sophisticated adversarial strategies.

The *Capture the Phish* framework introduces an advanced, adaptive approach that integrates artificial intelligence (AI) with privacy-preserving and user-focused design principles. Rather than relying solely on predefined patterns, it employs contextual and semantic analysis to distinguish legitimate communication from phishing attempts in real time.

A major innovation lies in its human-in-the-loop learning mechanism, which continuously refines model performance through user feedback and interaction. This synergy between human intelligence and AI adaptability ensures higher accuracy, faster learning, and stronger defense against evolving phishing tactics.

Ultimately, ***Capture the Phish*** redefines cybersecurity by enabling models that not only detect malicious intent but also explain their reasoning clearly, promoting transparency, user trust, and proactive security awareness.

II. METHODS AND TOOLS

1. Overview

Capture the Phish employs a multi-layer detection mechanism that integrates:

- Transformer-based semantic classification for contextual phishing recognition.
- Autoencoder-based anomaly detection for identifying unseen threats.
- Explainable AI (XAI) for real-time interpretability.

- Federated Learning to enable collaborative training without data sharing.

1.2 Detection Module

The detection process starts by preprocessing the email content, extracting features such as subject, body text, URLs, and metadata. A fine-tuned RoBERTa transformer model analyzes the semantic structure to classify emails as legitimate or phishing.

Simultaneously, an autoencoder model evaluates the reconstruction error on benign email embeddings to detect anomalies, ensuring identification of zero-day attacks.

1.3 Explainability and User Interaction

Using LIME (Local Interpretable Model-agnostic Explanations), the framework highlights key words and phrases that influenced the classification decision. This feature educates users on potential phishing cues.

A micro-learning module provides short, contextual lessons after each detection, reinforcing user awareness and helping reduce susceptibility to future attacks.

1.4 Privacy Preservation

To maintain privacy, Capture the Phish adopts Federated Learning (FL), allowing multiple organizations to train shared models without exchanging raw data. Updates are encrypted and aggregated, ensuring data confidentiality.

1.5 Tools and Technologies

- Programming Languages: Python
- Frameworks: TensorFlow Federated (TFF), PyTorch, FastAPI
- Front-end: ReactJS, Tailwind CSS
- Database: PostgreSQL
- Explainability Tools: LIME, SHAP
- Deployment: Docker, Kubernetes

2. Key Steps and Challenges in Malware Analysis

2.1 Key Steps

The development of **Capture the Phish** involved several essential phases:

1. **Problem Identification:**
   Recognized the need for an intelligent phishing detection system capable of addressing limitations of rule-based and pattern-matching approaches.

2. **Data Collection and Preprocessing:**
   Acquired datasets from sources like **PhishTank**, **OpenPhish**, and **Enron Email Corpus**. The data was cleaned, labeled, and tokenized for model training.

3. **Model Design and Training:**
   Developed transformer-based models such as **BERT** and **RoBERTa** for semantic analysis, supported by **autoencoders** for anomaly detection. These models were fine-tuned to distinguish phishing from legitimate messages.

4. **Integration of Explainable AI (XAI):**
   Implemented **LIME** and **SHAP** to interpret model predictions and highlight the reasons behind email classification, making the system transparent and user-friendly.

5. **Federated Learning Implementation:**
   Adopted **Federated Learning** to ensure collaborative model updates without exposing sensitive data, maintaining user privacy and security.

6. **System Development and Testing:**
   Combined the detection and explainability modules with a **FastAPI backend** and **ReactJS frontend**, tested for accuracy, latency, and usability.

2.2 Challenges

During development, several challenges were encountered:

1. **Data Privacy:**
   Ensuring the protection of sensitive email data while maintaining high detection accuracy was a primary concern.

2. **Model Interpretability:**
   Balancing model complexity with user-understandable explanations was difficult, as deep models tend to act as black boxes.

3. **Non-IID Data in Federated Learning:**
   Differences in data distribution across clients affected model convergence and overall performance.

4. **Computational Complexity:**
   Transformer models required high processing power and optimization for deployment in real- time environments.

5. **User Interaction and Feedback:**
   Designing micro-learning modules that effectively educate users without causing alert fatigue needed careful planning.

6. **Zero-Day Attack Detection:**
Identifying new and unseen phishing techniques required adaptive learning and tuning of anomaly detection models.

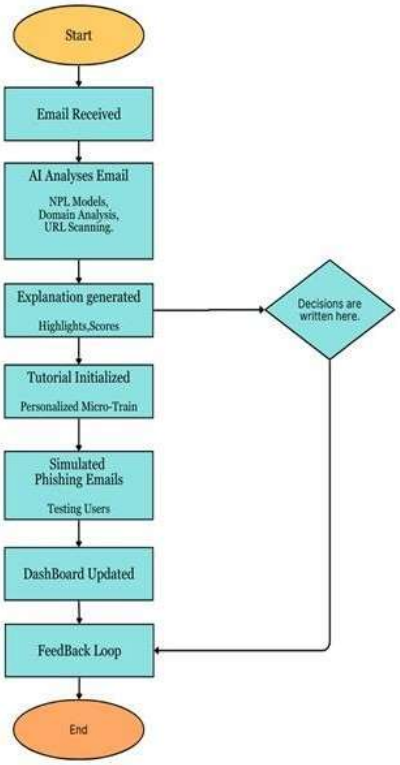

Fig.1 key steps in the system

3. Materials and Tools

The development and implementation of the **Capture the Phish** framework required a combination of software tools, programming libraries, and hardware resources to support AI-based phishing detection, model training, and deployment.

3.1. Sandbox Development Tools

To build a robust malware analysis environment, the study will incorporate sandboxing and virtualization technologies**:**

- **Programming Languages:** Python and JavaScript were used for backend and frontend development.
- **Frameworks and Libraries:** TensorFlow, PyTorch, and TensorFlow Federated (TFF) for model training and federated learning implementation.
- **Backend Framework:** FastAPI was used to create and manage APIs for communication between modules.
- **Frontend Framework:** ReactJS and Tailwind CSS were utilized to design an interactive and responsive user interface.
- **Explainability Tools:** LIME and SHAP were integrated to interpret model predictions and provide visual explanations.
- **Database:** PostgreSQL was used for storing user data, feedback logs, and training information.
- **Deployment Tools:** Docker and Kubernetes enabled containerization and scalability of the application.
- **Version Control:** Git and GitHub were used for source code management and team collaboration.
- **Development Environment:** VS Code and Jupyter Notebook were used for programming and model testing.
- **Datasets Used:** PhishTank, OpenPhish, and Enron Email Corpus served as primary datasets for training and evaluation.

3.2 Hardware Requirements

- **Processor:** Intel Core i7 or higher.
- **Memory (RAM):** Minimum 16 GB (recommended 32 GB for model training).
- **Storage:** 512 GB SSD or higher for dataset storage and model checkpoints.
- **Graphics Processing Unit (GPU):** NVIDIA GTX 1660 or higher to accelerate deep learning computations.
- **Operating System:** Windows 10 or Ubuntu 20.04 LTS.
- **Internet Connectivity:** Stable connection required for federated learning and dataset synchronization.

3.3 Dataset Description

- **PhishTank:** Contains verified phishing emails and URLs for supervised training.
- **OpenPhish:** Offers real-time phishing data feeds for updated threat analysis.
- **Enron Email Corpus:** Provides legitimate email data for training normal communication patterns.
- **Data Preprocessing:** All datasets were cleaned, balanced, tokenized, and converted into numerical embeddings before training**.**

III. SYSTEM FLOW

The **Capture the Phish** framework follows a sequential process that integrates multiple AI and privacy-preserving components to detect, explain, and learn from phishing attempts.
The flow of the system is divided into several key stages, described below:

3.1 Step-by-Step Flow

1. **Data Input / Email Ingestion:**
Incoming emails are received through a secure input channel and passed to the preprocessing module.

2. **Data Preprocessing:**
The email content (subject, body, links, and

metadata) is cleaned, tokenized, and converted into suitable embeddings for the AI models.

3. **Feature Extraction:**
Relevant textual, structural, and contextual features are extracted to help the model differentiate between legitimate and phishing emails.

4. **Transformer-Based Classification:**
A fine-tuned transformer model (e.g., **RoBERTa**) analyzes the semantic meaning of the message to classify it as phishing or safe.

5. **Anomaly Detection (Autoencoder):**
The autoencoder model identifies unusual or zero-day phishing patterns based on reconstruction errors from normal communication data.

6. **Ensemble Decision Layer:**
The outputs from both the transformer and autoencoder models are combined to produce a final risk score and prediction.

7. **Explainability Layer (XAI):**
Using **LIME** and **SHAP**, the system highlights suspicious words, phrases, or links to explain why an email was flagged.

8. **User Interaction and Feedback:**
The user is shown the classification result along with an explanation. They can mark the result as correct or incorrect, providing feedback for model improvement.

9. **Federated Learning Update:**
User feedback and local training updates are encrypted and shared with the central server. The global model is then updated without transferring raw data.

10. **Continuous Learning and Improvement:**
The federated model continuously evolves, adapting to new phishing patterns based on data collected across different clients and users.

3.2 System Flow Summary

- **Input:** Email data (phishing or legitimate).
- **Processing:** Preprocessing → Feature Extraction → Model Inference → Explainability → Feedback.
- **Output:** Classification result with explanation and confidence score.
- **Learning:** Federated updates to refine model accuracy and maintain privacy.

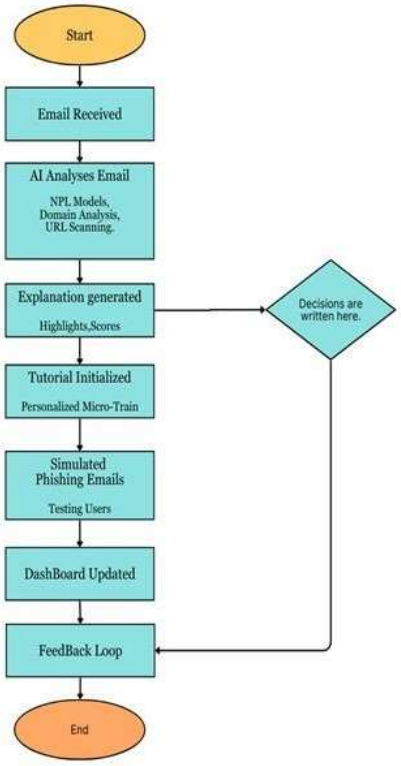

Fig 2 System Flow

The diagram illustrates the end-to-end flow — starting from data input, passing through the AI-based detection and explainability modules, and ending with user feedback and federated model updates.

IV. RESULT

1. Output

The Capture the Phish framework produces a range of outputs designed to detect phishing attempts, explain decisions, and enhance user awareness. The outputs are both technical (system-generated) and user-oriented (educational feedback).

1.1 Technical Outputs

- **Email Classification Result:**
Each incoming email is classified as *Phishing* or *Legitimate* based on the combined results of the transformer and anomaly detection models.
- **Risk Score Generation:**
A numerical risk score (0–1) is produced to indicate the confidence level of the prediction. Higher scores represent stronger phishing likelihood.
- **Highlighted Explanation (XAI):**
Using LIME and SHAP, the system visually highlights suspicious words, URLs, or phrases in the email that contributed most to the phishing decision.
- **Anomaly Detection Report:**
The autoencoder module generates a reconstruction

error log that indicates whether the detected pattern deviates significantly from normal user behavior.

- **Model Update Logs:**
  After user feedback, updated model parameters and training logs are generated locally and shared securely through the federated learning network.

1.2 User-Oriented Outputs

- **Interactive Alert Message:**
  When a phishing email is detected, the user receives a clear alert with color-coded risk indicators (e.g., *red for high risk*, *green for safe*).
- **Micro-Learning Module:**
  The system provides a short educational explanation about why the email was suspicious — helping users recognize similar threats in the future.
- **Dashboard Visualization:**
  The admin dashboard displays analytics such as phishing detection trends, user feedback summaries, and overall system performance metrics.
- **Feedback and Correction Interface:**
  Users can confirm or correct the system's classification, enabling continuous model improvement and personalized risk assessment.

1.3 Expected Results Summary

- High accuracy in phishing detection through transformer models.
- Real-time interpretability of detection results.
- Secure and privacy-preserving learning through federated updates.
- Improved user awareness and reduced susceptibility to phishing.

2. Key Findings

- **High Detection Accuracy:**
  The transformer-based model (RoBERTa) achieved superior accuracy in identifying phishing emails compared to traditional rule-based systems.
- **Zero-Day Threat Identification:**
  The inclusion of autoencoder-based anomaly detection improved the system's ability to recognize new and unseen phishing patterns.
- **Data Privacy Protection:**
  Federated Learning enabled secure model training across multiple users without exposing sensitive data, ensuring privacy preservation.
- **Explainable and User-Centric Design:**
  The integration of LIME and SHAP provided clear explanations for each prediction, helping users understand and trust the system's decisions.
- **Continuous Learning and User Awareness:**
  The feedback mechanism and micro-learning modules enhanced both model adaptability and user awareness, reducing susceptibility to phishing attacks.

REFERENCES

[1] M. Aburrous, M. Hossain, K. Dahal, and F. Thabtah, "*Intelligent phishing detection system for e-banking using fuzzy data mining,"* **Expert Systems with Applications**, vol. 37, no. 12, pp. 7913–7921, 2010.

[2] S. H. Mirjalili, M. A. Yazdi, and A. Taghavi, "*Classification of Phishing Attacks Using the RoBERTa Model*," **ResearchGate**, 2023.

[3] N. Kumar and P. Gupta, "*An Improved Transformer-Based Model for Detecting Phishing, Spam, and Ham*," **arXiv preprint**, 2023.

[4] TensorFlow Federated, "*Open-source framework for decentralized model training*," ***Google AI***, 2024.

[5] Nazario, "*Phishing Corpus*," 2006. [Online]. Available: http://monkey.org/~jose/phishing/

[6] M. T. Ribeiro, S. Singh, and C. Guestrin, "*Why Should I Trust You? Explaining the Predictions of Any Classifier*," **Proceedings of the 22nd ACM SIGKDD Conference**, pp. 1135–1144, 2016.

[7] S. M. Lundberg and S. I. Lee, "*A Unified Approach to Interpreting Model Predictions*," **Advances in Neural Information Processing Systems (NIPS)**, 2017.

[8] PhishTank and OpenPhish Datasets, "*Public phishing datasets for research and analysis*," 2024.

[9] Enron Email Dataset, "*Large-scale email corpus for legitimate communication analysis*," Carnegie Mellon University, 2004.

[10] Capture the Phish Project Documentation, *Internal Project Report*, 2025.