CS-3610: Information Security

Subhashis Banerjee, Ashoka Univeristy

Project-cum-Assignment

Neha Palak 1020231795

November 15, 2025

Designing a Multi-Layered Anti-Phishing System to Mitigate Malicious Emails and Links

Contents

1 Introduction

Phishing is one of the most persistent and damaging cyber threats. Attackers imitate trusted organizations to trick users into clicking malicious links or revealing sensitive information such as passwords or financial details. Recent reports indicate that over 90% of successful cyberattacks start with a phishing email.

While email providers use spam filters, these mainly target bulk unsolicited mail and often miss targeted or sophisticated phishing that relies on social engineering and domain spoofing. The challenge is detecting such threats without compromising usability or privacy.

This project proposes a multi-layered anti-phishing system that combines several security mechanisms to detect, block, and report malicious emails and links. A prototype implementation (`scan_links.py`) demonstrates the content and link analysis component of the design.

2 Problem Statement

Users and organizations continue to be vulnerable to malicious emails and embedded links that slip past traditional filters. Attackers keep improving their methods by using URL shorteners, visually confusing domains (like "paypa1.com"), and HTTPS certificates that make fake sites appear trustworthy.

Because there is no complete, end-to-end security system that protects every stage—from validating the sender to guiding the user's actions—phishing attacks can still succeed even when individual defenses are in place.

This project aims to design and demonstrate a holistic defense architecture that addresses these gaps by combining several independent layers:

- Authentication protocols to validate sender legitimacy.
- Automated content and link analysis to inspect potential threats.
- User interface warnings to guide user behavior.
- Reporting and monitoring mechanisms to support continuous learning.

3 Threat Model

3.1 Threat Actors and Adversaries

Threat Actors:

- Email Service Provider (institutional mail servers)
- IT/Security Team
- End Users (students, faculty, staff)
- External Legitimate Senders
- Email Clients (web/mobile apps)
- Network Infrastructure (DNS, routers)
- SPF/DKIM/DMARC record publishers
- Prototype components (scanner, heuristics, blocklist)

Adversaries:

- External attackers/cybercriminals
- Compromised legitimate senders
- Malicious insiders
- Compromised DNS resolvers or network nodes
- Third-party adversaries (phishing-kit operators, botnets, domain squatters)

3.2 Trust and Verifiability

Trust:

- Domain owners publishing SPF/DKIM/DMARC records
- Organization's IT/security team
- Legitimate external mail senders

Verfiability:

- Sender domain
  - Threat: spoofing
  - Mitigation: SPF/DKIM/DMARC
- URLs in email body
  - Threat: obfuscated domains / shortened links
  - Mitigation: heuristics + blocklist
- Displayed link text vs. actual href
  - Threat: mismatch to mislead users
  - Mitigation: UI preview + mismatch detection
- DNS/HTTPS resolution
  - Threat: redirection or poisoned DNS
  - Mitigation: secure DNS + HTTPS validation

3.3 Threats

- Domain spoofing by external attackers
  - Goal: impersonate trusted entities
- Malicious or obfuscated URLs
  - Goal: steal credentials or deliver malware
- Display-text vs. href mismatch in HTML emails
  - Goal: deceive user about true destination
- Compromised insider or legitimate sender account
  - Goal: targeted spear-phishing with high trust

3.4 Mitigations

- SPF/DKIM/DMARC authentication
- Heuristic URL scanning + blocklist matching
- UI-based warnings and link previews
- Logging, monitoring, and “Report Phishing” workflows

4 Proposed Solution

The proposed system follows a defense-in-depth approach, where multiple independent layers collectively ensure end-to-end protection.

4.1 Layer 1: Email Authentication

To prevent spoofing, the system enforces three widely adopted email authentication standards:

- SPF (Sender Policy Framework): Verifies that an email is sent from an authorized mail server.
- DKIM (DomainKeys Identified Mail): Uses cryptographic signatures to verify message integrity.
- DMARC (Domain-based Message Authentication, Reporting and Conformance): Aligns SPF and DKIM checks with domain policies and provides reporting capabilities.

Emails failing these checks are quarantined or marked as unverified. This layer filters out most impersonation-based attacks before they reach users.

4.2 Layer 2: Content and Link Analysis

This layer inspects the body of each email for suspicious content or embedded links.

The implemented prototype, `scan_links.py`, demonstrates this functionality through:

- URL extraction using regex-based scanning.
- Blocklist matching against known malicious domains.
- Heuristic detection of obfuscated or shortened URLs.
- Pattern recognition for misleading keywords such as “verify,” “account,” or “urgent.”

These checks run automatically when new emails are received. URLs that match suspicious patterns are flagged for review.

4.3 Layer 3: User Interface and Warning Design

Even the most advanced detection mechanisms can fail if users ignore warnings. This layer focuses on human-centric security, integrating features such as:

- Warning banners for external or unverified senders (“ This message is from outside your organization”).
- Color-coded link previews that highlight mismatched or shortened URLs.
- “Report Phishing” buttons in the email interface that forward suspicious messages to the security team.

By informing users in real time, the system reduces reliance on perfect automated detection.

4.4 Layer 4: Reporting and Monitoring

Every flagged email or reported phishing attempt is logged and analyzed to improve the detection model.
Logs contain metadata such as sender domain, URL patterns, and timestamp (without storing message content to preserve privacy). Periodic summaries help update the blocklist, tune heuristics, and train users on new phishing trends.

5 Prototype Implementation

The prototype demonstrates how simple, interpretable heuristics can flag suspicious URLs.

5.1 Components

- `scan_links.py`: Python script implementing the analysis logic.
- `blocklist.txt`: Simulated repository of known malicious domains.
- `sample_emails/`: Folder containing three sample email files representing phishing, benign, and borderline cases.

5.2 Functionality

The script extracts all URLs from a text file, checks them against the blocklist, and applies pattern-based heuristics (like detecting "0fficial" or numeric domain patterns).

Example Output:

```
  URL: https://bit.ly/verify-paypal-12345
- FLAGGED: shortened URL (obscures target)
- FLAGGED: matches obfuscation heuristic ((?i)paypal)
```

5.3 Design Choice

The implementation intentionally uses simple, transparent logic rather than a complex ML model to emphasize explainability and auditability: two key requirements in real-world email security systems.

6 Design Rationale

Security is not achieved through a single mechanism but through a combination of complementary controls.

- Layered Defense: Each layer mitigates specific attack vectors; if one fails, others still offer protection.
- Integration of Technical and Human Elements: Automated scanning detects most phishing attempts, while user training and reporting handle edge cases.
- Usability and Privacy Considerations: Minimal false positives reduce user fatigue, and all analysis is performed locally or within the organization's infrastructure.
- Scalability: The modular design allows integration with enterprise email gateways or cloud-based APIs.

This approach aligns with security engineering principles such as *least privilege, defense in depth*, and *security by design.*

7 Evaluation

The prototype was evaluated using three representative email samples to test its ability to detect malicious links while avoiding false positives.

Email File	Type	Expected Outcome	Actual Result
email1.txt	Phishing (shortened link)	Flagged for obfuscation and URL shortener	Correctly flagged
email2.txt	Benign newsletter	No warnings or flags	Correctly ignored
email3.txt	Suspicious domain	Flagged via blocklist or heuristics	Correctly flagged

Table 1: Evaluation Summary

The results show that the system reliably detects common phishing indicators such as shortened URLs, obfuscated patterns, and blocklisted domains, while keeping false positives low. This demonstrates the effectiveness of simple, interpretable heuristics as part of a multi-layered anti-phishing approach.

In real-world deployment, detection could be strengthened through live reputation APIs, sandboxed link inspection, ML-based classification, and continuously updated blocklists and heuristics.

8 Limitations and Future Scope

While effective for demonstration, this prototype and architecture have several limitations:

1. Static Heuristics: The current detection logic uses fixed patterns and a static blocklist; real systems must continuously update threat intelligence.
2. Attachment Scanning: The implementation focuses only on URLs, not file attachments or embedded scripts.
3. Limited Authentication Simulation: SPF/DKIM/DMARC layers are described conceptually but not implemented in code.
4. User Awareness: The design assumes users read and respond to warnings appropriately, something that often requires training and culture-building.

Future work could integrate:

1. Machine learning models for adaptive phishing detection.
2. Browser extensions for real-time link inspection.
3. Automated phishing report collection and visualization dashboards.

9 Conclusion

This project presents a comprehensive anti-phishing framework that combines multiple layers of security, email authentication, link analysis, user interface warnings, and continuous monitoring. The prototype demonstrates the feasibility and value of lightweight heuristic analysis for identifying malicious links within emails.

By integrating both technical mechanisms and human-centric design, the system offers a holistic defense against phishing attacks, aligning with modern information security principles and the end-to-end security objectives of this course.

10 References

1. https://www.ncsc.gov.uk/guidance/phishing
2. https://www.mimecast.com/content/dkim-spf-dmarc-explained/
3. https://www.sciencedirect.com/science/article/pii/S0167404823002973
4. https://www.ijisrt.com/assets/upload/files/IJISRT24FEB1107.pdf

11 Honour Code Statement

I affirm that this project is my own work. Generative AI was used only to gain insights on certain topics and to help draft parts of the conclusion and problem statement; all research, analysis, design decisions, coding, and final revisions were done independently.