ICT30016 - ASSIGNMENT 2 : INNOVATION CONCEPT

Topic: Cybersecurity – Detection of malicious emails

Mentor: Baker Alrubaiey

Email: balrubaiey@swin.edu.au

06-MAY-2024

Thang Truong - 103422254@student.swin.edu.au

Debra Chirchir - 105465125@student.swin.edu.au

LachIan James - 101362125@student.swin.edu.au

Teisha Damman - 104539016@student.swin.edu.au

Executive summary

In today's world, there is no doubt that people can access the internet across a variety of devices, including smart watches, mobile phones, tablets, and laptops, from anywhere. This widespread accessibility is a remarkable advantage in terms of technological innovation. However, this widespread accessibility also presents challenges in the cybersecurity industry. Malicious actors exploit anonymity to target internet and email users by deploying malware-infected and phishing emails, employing clever and innovative strategies to evade anti-spam filters[1], [2].

We have conducted extensive research and developed innovative design concepts related to methodology, specification, vulnerability analysis, and justification. Our goal is to assist email users in distinguishing between malicious and legitimate emails. For instance, we utilize machine learning techniques, including Convolutional Neural Networks (CNN) for malware detection and Random Forests (RF), combining predictions from multiple models to enhance accuracy in terms of Malware infected emails [3] . And another method is one remarkable approach, put forth by Yasin and Abuhasan (2016) [8], leverages linguistic processing methods and ontologies to improve the semantic similarity between emails in terms of Phishing emails. Moreover, critical elements within Spear Phishing Defence systems, such as Email Parser, Threat Detector, and the Quarantine Mechanism, excel at efficiently detecting and neutralizing spear phishing attempts [9]. We firmly believe that our design concepts can assist internet and email users in distinguishing between malicious and legitimate emails.

Table of Contents

Task allocation

1) Thang Truong
    a. Proposal template
    b. Executive Summary
    c. Project overview
    d. Client requirements
    e. Review and combine tasks
    f. Recommendation
2) Debra Chirchir - Design concept 1 (Malware Infected Emails) which included (Methodology, Design constraints, Specifications Vulnerability analysis and Justification of your design).
3) LachIan James - Design concept 2 (Phishing Emails) which included (Methodology, Design constraints, Specifications Vulnerability analysis and Justification of your design).
4) Teisha - Design concept 3 (Spear Phishing Defence systems) which included (Methodology, Design constraints, Specifications Vulnerability analysis and Justification of your design)

Part A

Project overview

Objective

The project aims to assist individuals and enterprises globally in identifying and classifying malicious emails. Additionally, it involves analyzing collected data to uncover the common characteristics of such emails. Finally, the project report will recommend cutting-edge algorithms that leverage Machine Learning and AI to effectively filter out malicious emails and enhance user security.

Scope

The report refrains from introducing novel algorithms, recommending subscriptions with third-party agencies, or endorsing the purchase of the latest market tools for filtering out malicious emails. Instead, it relies on information cited from recent articles, which will be listed in the references section below.

Goals

Our goal is to substantially enhance the detection and mitigation of malicious emails for individuals, businesses, and organizations worldwide. Given the evolving cyber threats, robust email security measures are essential to safeguard sensitive information and prevent data breaches.

Client requirements

Based on our discussions and viewpoint, there are certain shared client expectations that need to be met consistently and effectively. These expectations fall into the following categories [1], [2].

- Accuracy
    - The new algorithms need to achieve high accuracy in identifying malicious emails for comparison with previous algorithms while also

minimizing false positives to prevent legitimate emails from being incorrectly flagged as malicious.

- Reporting and Alerts:
  - Create an intuitive interface that generates comprehensive reports on identified malicious emails and promptly alerts administrators or end-users when suspicious emails are detected.
- Real-Time Scanning
  - Productivity and Effectively in terms of detection and response are critical.
- Scalability and Infrastructure
  - The new algorithm's ability to handle a large volume of incoming emails efficiently and considered for future growth.
  - Ensure seamless integration with the client's existing email servers or clients.
- Customizable Rules and Performance metrics
  - Empower the client to establish personalized rules for email classification such as according to keywords, sender reputation, attachment types, etc.
  - Consistently assess and provide updates on system performance
- Testing and Evaluation
  - Comprehensively validate the solution by testing it with diverse datasets, including both malicious and legitimate emails. For instance, consider using tools like KNIME to obtain statistical insights. This tool can evaluate performance metrics (accuracy, precision, recall) to validate effectiveness.
- Privacy
  - Safeguard user privacy during the analysis of email content.

Part B

Design concept 1: Malware Infected Emails

In the design concept for a system to handle malware-infected emails, the documented contributions include establishing a version-controlled repository to track changes and facilitate collaboration. Coding standards are set to ensure readability and maintainability, while commenting standards are implemented to provide clarity on code functionality. The environment setup is tailored to replicate production conditions closely, allowing for accurate testing. The prototype architecture is modular, with components such as an email parser, a threat detector, and a quarantine mechanism [5].

Design Concept - Explanation

Design Concept	Explanation
Repository Setup	This involves creating a central location where all the code is stored and managed. It includes setting up a version control system like Git, which allows multiple developers to work on the project simultaneously without overwriting each other's changes. The repository should also include a README file that provides an overview of the project, installation instructions, and how to contribute.
Coding Standards	These are a set of guidelines for writing code. They can include rules for naming variables and functions, indentation, use of spaces or tabs, maximum line length, and so on. The goal of coding standards is to maintain consistency, which makes the code easier to read and understand.

Commenting Standards	Comments are used to explain what a particular piece of code does, which is especially useful for complex or non-obvious code. A good commenting standard might include rules for when to comment, what information to include in comments, and the format of the comments.
Environment Setup	This involves setting up the development environment needed to work on the project. It can include installing the necessary software, setting up databases, configuring servers, and so on. It's important to document this process so that new developers can get up and running quickly.
Prototype Architecture and Components	This refers to the high-level structure of the software. It includes the major components of the software, how they interact with each other, and the design patterns used. Documenting the architecture can help developers understand the big picture and where their work fits in.

Preliminary design

Methodology

- Design Solution 1 - User-Centered Design (UCD): This approach involves designing with a deep understanding of the user and their needs at the forefront. The essential characteristics of the design problem are retained by ensuring that the solution is tailored to the user's needs. The methodology involves several stages:
    - User Research: Understand the user's needs, behaviors, and pain points through methods like interviews, surveys, and user testing.
    - Ideation: Generate a wide range of ideas to solve the user's problems.

- Prototyping: Create low-fidelity prototypes of the design solution.
- User Testing: Test the prototypes with users to gather feedback.
- Iteration: Refine the design based on user feedback and repeat the process until the solution meets the user's needs.

- Design Solution 2 - Agile Design: This approach involves iterative development where requirements and solutions evolve through collaboration. It retains the essential characteristics of the design problem by allowing for flexibility and adaptability in the design process. The methodology involves:

  - Planning: Define the problem and plan the solution.
  - Design & Development: Design and develop the solution in small, manageable increments.
  - Testing & Review: Regularly test and review the solution to ensure it meets the requirements.
  - Iteration: Make necessary changes and improvements based on feedback and repeat the process.

- Design Solution 3 - Systems Design: This approach views the design problem as part of a larger system and seeks to design solutions that fit seamlessly within this system. It retains the essential characteristics of the design problem by considering the broader context in which the problem exists. The methodology involves:

  - System Analysis: Understand the larger system in which the design problem exists.
  - Design: Create a solution that fits within this system and addresses the problem.

- Implementation: Implement the design in the larger system.
- Evaluation: Evaluate the effectiveness of the design within the system and make necessary adjustments

Each of these methodologies employs a different approach to the design process, but all aim to create effective and user-friendly design solutions that address the essential characteristics of the design problem. They all involve a cycle of planning, design, implementation, and evaluation, with the goal of continuous improvement and adaptation to changing user needs and contexts.

- Design constraints:

Constrains	Issues
Economic Constraints	These are related to the budget and resources available for the design. They can influence the choice of materials, the complexity of the design, and the time allocated for the project.
Legal Constraints	These are the laws and regulations that the designer must comply with. They can include safety standards, privacy laws, accessibility standards, and industry-specific regulations.
Time Constraints	These are related to the schedule and deadlines for the project. They can influence the scope of the project, the complexity of the design, and the prioritization of different aspects of the design.

Each of these constraints plays a significant role in shaping the final design concept. They require the designer to make trade-offs and prioritize certain

aspects of the design over others. Understanding these constraints is crucial for developing a successful and effective design solution

- Specifications: Outline the technical specifications and requirements specific to this design concept.
    - Security: The system should adhere to best practices for cybersecurity. This could involve specifications for encryption, user privacy, secure connections, and compliance with data protection regulations.
    - Functionality: The system should be able to detect and quarantine malware-infected emails. This could include specific features like scanning attachments, analyzing email content for malicious links, and isolating infected emails.
- Vulnerability analysis: Conduct an analysis of potential vulnerabilities in the design and discuss measures to address them [4].

Phishing Attacks	Phishing is a common method used by attackers to trick users into revealing sensitive information. If the system fails to accurately identify phishing emails, users could be at risk. Mitigation Measures: Implement advanced phishing detection algorithms that can analyze the content of emails and identify potential phishing attempts. Regularly update the system with the latest known phishing strategies.
Zero-Day Threats	These are new, previously unknown threats that can exploit vulnerabilities in the system before they are identified and patched. Mitigation Measures: Regularly update the system's threat database and

	use machine learning algorithms to predict and identify potential zero-day threats.
Spoofing Attacks	Attackers may attempt to spoof email addresses to appear as a trusted source, bypassing filters. Mitigation Measures: Implement robust email authentication protocols like SPF, DKIM, and DMARC to verify the authenticity of emails
User Error	Users might accidentally whitelist malicious email addresses or click on malicious links in emails
Encryption-Based Threats	Some sophisticated malwares can be hidden in encrypted attachments or links, making them harder to detect.

Justification of design

Here are three innovative design concepts that leverage machine learning to improve upon existing algorithms:

Deep Learning for Malware Detection: Deep learning algorithms [3], such as Convolutional Neural Networks (CNN), can be used to detect malware in emails. These algorithms can learn complex patterns in data, making them effective at detecting sophisticated malware attacks. They can be trained on a large dataset of emails, learning to distinguish between benign and malicious emails based on their content and structure.

Ensemble Learning for Improved Accuracy: Ensemble methods, such as Random Forests (RF), combine the predictions of multiple machine learning models to make a final decision [4]. This approach can improve the accuracy of malware detection by leveraging the strengths of different algorithms. For example, one model might be good at detecting phishing emails, while another

might excel at spotting ransomware. By combining their predictions, the system can effectively detect a wide range of threats.

Active Learning for Handling New Threats: Active learning is a semi-supervised machine learning technique where the model actively queries the user for labels in instances where it is uncertain. This approach can be particularly useful in the context of email malware detection, where new threats are constantly emerging. When the system encounters an email that it's unsure about, it can ask a security analyst for a label, and then use this information to update its model. This allows the system to continually adapt to new threats [5].

Each of these design concepts offers a novel and logical approach to email malware detection, drawing upon the latest research in machine learning. They each have their strengths and could potentially be combined to create a highly effective, multi-faceted defense against email-borne malware.

Design concept 2: Phishing Emails

Our project on phishing email detection is driven by the goal of establishing a robust and effective system for identifying and mitigating email-based security threats [6]. In pursuit of this objective, we recognize the critical importance of meticulously planning and executing the setup and development process. Our approach prioritizes numerous key aspects to ensure the success and sustainability of our implementation.

Design concept - Explanation

- Comprehensive Repository Management

We established a centralized repository using industry-standard version control systems – in this case GitHub. This repository serves as the backbone

of our collaborative development efforts, providing a unified platform for team members to share, review and contribute to the project codebase. By adhering to the best practices in repository management, including branch management strategies and pull request workflows, we foster an environment of transparency, accountability, and version control integrity.

- Adherence to Coding Standards and Guidelines

Consistent adherence to coding standards and guidelines is fundamental to the maintainability, readability, and scalability of our codebase. We embrace established coding conventions to ensure uniformity and clarity in our code. Additionally, we prioritise the use of meaningful variable names, modular code organization, and proper documentation practices to enhance code comprehension and facilitate collaborative development efforts.

- Thorough Commenting Standards

Clear and informative code documentation is crucial for promoting readability and facilitating knowledge sharing among team members. We adopt a rigorous approach to commenting standards, ensuring that each function, class, and module is accompanied by descriptive comments outlining its purpose, inputs, outputs, and relevant implementation details. Consistent commenting practices empower developers to understand, modify and extend the codebase confidently and clearly.

- Seamless Environment Setup

Creating a dependable and consistent development environment is essential for efficient setup and uniformity across diverse environments. We utilize containerization to encapsulate project dependencies, simplifying the creation of isolated development environments. Automated provisioning tools further

reduce manual tasks, ensuring consistency and predictability for all team members.

- Scalable Prototype Architecture and Components

The prototype architecture encompasses a multi-layered approach to phishing email detection, comprising data reprocessing, feature extraction, classification, and response mechanisms. Components such as natural language processing techniques, machine learning modules, and email filtering algorithms are integrated to analyse email content, headers, and attachments for indicators of phishing behaviour.

Preliminary design

Methodology: The methodology adopted for the detection of phishing emails revolves around the application of Knowledge Discovery (KD) and data mining techniques [7]. The primary Objective is to develop an efficient email classifier capable of accurately categorizing incoming emails as either spam or legitimate. The process commences with the implementation of KD steps, which involves extracting pertinent features from a training dataset comprising email samples. These extracted features serve as input for a variety of data mining algorithms, aimed at identifying the most effective classifier. One notable model, as proposed by Yasin and Abuhasan (2016) [8], employs linguistic processing techniques and ontologies to enhance the semantic similarity among emails. This enhancement contributes to the overall performance and efficiency of the classification process, leading to improved accuracy in distinguishing between spam and legitimate emails. Furthermore, the selection of papers for evaluation was meticulously conducted, focusing on research works that demonstrate significant impact and intelligent automation in phishing email detection. These selected papers were thoroughly analysed

to assess the utilization of machine learning principles, the robustness and effectiveness of the proposed solutions and the necessary modifications required to address any identified drawbacks.

Design Constraints

- Data availability

The availability and quality of labelled phishing and legitimate email datasets can significantly impact the training and testing phases of the detection program. Constraints related to data collection, labelling and preprocessing must be addressed to ensure the program's accuracy and generalization.

- Scalability

The ability of the detection program to scale with increasing email volume and diversity (Multiple languages) is essential for deployment in large-scale email systems. Designing scalable algorithms and infrastructure to manage growing data volumes and user traffic without compromising performance is a significant constraint.

- Regulatory compliance

Compliance with data protection regulations, privacy laws, and industry standards may impose constraints on data handling, storage, and processing within the detection program. Ensuring compliance with relevant regulations and standards is essential for legal and ethical use of the program.

Specifications

- Email collection and preprocessing

Implement email collection mechanisms to retrieve messages in real-time. Preprocesses emails to extract relevant metadata and content for analysis.

- Feature extraction and selection

Implement feature extraction techniques, such as keyword frequency, header analysis, and link properties. Select relevant features to optimize detection accuracy and minimize computational overhead.

- Machine learning models

Implement supervised learning algorithms, such as logistic regression, decision trees, or neural networks, to train classification models.

- Real time detection

Implement streaming algorithms or event driven architectures to process incoming emails in real-time. Utilize parallel processing and distributed and computing to manage high email volumes efficiently.

- Alerting and Reporting

Implement alerting mechanisms, such as email notifications or dashboard alerts to notify stakeholders of suspicious emails. Generate detailed reports on phishing activity, including statistics, trends, and remediation actions.

- Integration and Compatibility

Design APIs or integration points to connect the phishing email detection system with email servers, firewalls, Security information and event management (SIEMs) and other security solutions. Support standard email protocols and interoperability with major email clients and platforms.

- Adaptability and Updates

Implement mechanisms for continuous monitoring and updating of detection algorithms and threat intelligence feeds. Incorporate feedback loops to learn from detected phishing attempts and improve future detection capabilities.

Vulnerability Analysis

- Vulnerability one

The system may incorrectly classify legitimate emails as phishing or fail to detect actual phishing emails.

Address this by Implementing engineering techniques to enhance the accuracy of feature extraction, continuously update and refine machine learning models using feedback loops.

- Vulnerability two

Sophisticated attackers may employ evasion techniques to bypass detection. Utilize advanced content analysis algorithms such as natural language processing and image recognition. Regularly update threat intelligence feeds

Justification of design

The design specifications for the phishing detection system address crucial aspects for success and alignment with objectives. Implementing real-time email collect and preprocessing enables prompt identification of phishing threats, supporting proactive security measures. Feature extraction techniques optimize detection accuracy while minimizing computational overhead. Integration of supervised learning algorithms facilitates high-accuracy identification of phishing emails, leveraging advanced technologies for enhanced security. Real time detection ensures prompt mitigation of attacks, while alerting mechanisms and detailed reporting facilitate informed decision-making, aligning with the objective of providing actionable insights into threats. Designing for integration and compatibility maximizes usability and effectiveness. Overall, these specifications contribute to a robust system, aligning with objectives to mitigate email-based security threats.

Design concept 3: Spear Phishing Defence System

Design Concept Explanation

Spear phishing attacks pose a significant threat to organizations, requiring robust defence systems to mitigate risks effectively. The design concept for a Spear Phishing Defence System encompasses various components aimed at detecting and neutralizing spear phishing attempts efficiently.

Establishing a version-controlled repository is paramount for the Spear Phishing Defence System. A centralized repository, utilizing version control systems like Git, ensures collaborative development while tracking changes seamlessly. Including comprehensive documentation within the repository aids in understanding project overview, installation procedures, and contribution guidelines.

Adherence to coding standards guarantees the reliability and maintainability of the Spear Phishing Defence System's codebase. Clear guidelines on naming conventions, indentation, and documentation ensure consistency across the code, facilitating readability and ease of maintenance.

Implementing commenting standards enhances code comprehension and maintainability. Clear and concise comments elucidate the functionality of code segments, especially in complex modules or algorithms. Establishing rules for consistent comment formatting and content ensures clarity and comprehensiveness throughout the codebase.

Creating a dependable and consistent development environment is essential for efficient setup and uniformity across diverse environments. Utilizing containerization to encapsulate project dependencies simplifies the creation of isolated development environments. Automated provisioning tools further

reduce manual tasks, ensuring consistency and predictability for all team members.

The prototype architecture of the Spear Phishing Defence System comprises modular components designed to detect and mitigate spear phishing attacks effectively. Key components include:

Email Parser: Responsible for parsing incoming emails and extracting relevant content for analysis.

Threat Detector: Utilizes advanced algorithms to analyse email content, identify suspicious patterns, and detect potential spear phishing attempts.

Quarantine Mechanism: Implements measures to quarantine suspicious emails, preventing them from reaching end-users and mitigating potential risks.

Preliminary design

Methodology: To address the spear phishing threat comprehensively, the Spear Phishing Defence System employs a multi-faceted approach to design and development. The methodologies employed include

- User-Centered Design (UCD): Prioritizing user needs and behaviours ensures that the system effectively addresses the challenges posed by spear phishing attacks. Through stages such as user research, ideation, prototyping, user testing, and iteration, the system is tailored to meet user requirements and preferences.
- Agile Design: Embracing iterative development allows for flexibility and adaptability in responding to evolving spear phishing threats. By planning, designing, developing, testing, and iterating in incremental cycles, the system can quickly adapt to changing requirements and user feedback.

- Systems Design: Viewing the Spear Phishing Defence System as part of a larger cybersecurity ecosystem ensures seamless integration and interoperability. System analysis, design, implementation, and evaluation are conducted with a holistic perspective, considering the broader context in which the system operates.

Design Constraints

- Data Availability: The availability and quality of labelled phishing and legitimate email datasets can significantly impact the training and testing phases of the detection program.
- Scalability: The ability of the detection program to scale with increasing email volume and diversity (multiple languages) is essential for deployment in large-scale email systems.
- Regulatory Compliance: Compliance with data protection regulations, privacy laws, and industry standards may impose constraints on data handling, storage, and processing within the detection program.

Understanding and addressing these constraints are crucial for ensuring the successful design and implementation of the Spear Phishing Defence System while balancing technical, regulatory, and resource considerations.

Specifications

The Spear Phishing Defence System must meet specific technical specifications to effectively combat spear phishing attacks, including [9]:

- Advanced Threat Detection: Implementing state-of-the-art algorithms and techniques for analysing email content, detecting suspicious patterns, and identifying potential spear phishing attempts.

- Real-Time Response: Ensuring timely detection and response to spear phishing attacks, including automatic quarantine of suspicious emails and alert notifications to administrators.
- Integration Capabilities: Seamless integration with existing email infrastructure, security systems, and threat intelligence platforms to enhance overall cybersecurity posture.

Vulnerability Analysis

Conducting a vulnerability analysis is crucial for identifying potential weaknesses in the Spear Phishing Defence System and implementing measures to address them effectively. Key vulnerabilities to consider include:

- Email Spoofing: Attackers may attempt to spoof email addresses to impersonate trusted sources, bypassing email filters and security mechanisms.
- Social Engineering Tactics: Sophisticated social engineering tactics may deceive users into divulging sensitive information or performing malicious actions.
- Zero-Day Exploits: New and previously unknown vulnerabilities may be exploited by attackers to bypass security defences and launch spear phishing attacks undetected.

Mitigating these vulnerabilities requires a combination of robust security measures, ongoing monitoring, and proactive threat intelligence to stay ahead of emerging threats.

Justification of design

The design of the Spear Phishing Defence System incorporates advanced methodologies, stringent coding standards, and comprehensive specifications to address the evolving threat landscape posed by spear phishing attacks. By

leveraging modular architecture, user-centered design principles, and agile development methodologies, the system aims to provide a robust defence against spear phishing threats while accommodating constraints and specifications inherent to the project.

Recommendation and Conclusion

We wholeheartedly believe that our report can offer valuable support to everyone worldwide. It provides an overview of our innovative design concepts aimed at enhancing the detection of malicious emails for internet and email users. To safeguard against such threats, users should combine traditional and modern practices. However, the challenge lies in the fact that anonymous threat actors continually devise sophisticated strategies to evade the latest design innovations. From our research, we recommend that each user exercise caution by refraining from accessing addresses and promotions recommended by non-official emails. Simultaneously, businesses and organizations ought to deploy robust filters or rely on trusted applications to minimize the impact of malicious emails.

References

[1] Australian Government, (2021, October). Malicious Email Mitigation Strategies. https://tinyurl.com/yc45k5x4.

[2] Australian Government (2023, April). Protect yourself from malicious email. https://tinyurl.com/36mc3pdn.

[3] Saeed, U., Javed, A., Rehman, A., Hafeez, A., & Ali, W. (2022, September). Real-Time Forest Fire Detection Using Wireless Sensor Networks: A Comprehensive Review. Symmetry, 14(11), 2304. https://www.mdpi.com/2073-8994/14/11/2304

[4] Smith, J., & Johnson, A. (2023, March). Malware-infected emails and websites: A comprehensive analysis. Journal of Cybersecurity, 8(2), 123-135.

[5] Garcia, M., & Patel, R. (2024, March). Understanding the threat landscape: Malware-infected emails and websites. Cybersecurity Today, 6(3), 45-56.

[6] Karim, A., Azam, S., Shanmugam, B., Kannoorpatti, K., & Alazab, M. (2019). A comprehensive survey for intelligent spam email detection. IEEE Access, 7, 168261-168295. https://doi.org/10.1109/access.2019.2954791

[7] Salloum, S., Gaber, T., Vadera, S., & Shaalan, K. (2022). A systematic literature review on phishing email detection using natural language processing techniques. IEEE Access, 10, 65703-65727. https://doi.org/10.1109/access.2022.3183083

[8] Yasin, A., & Abuhasan, A. (2016). An intelligent classification model for phishing email detection. International Journal of Network Security & Its Applications, 8(4), 55-72. https://doi.org/10.5121/ijnsa.2016.8405

[9] Catal, C., Giray, G., Tekinerdogan, B., Kumar, S., & Shukla, S. (2022, May). Applications of deep learning for phishing detection: A systematic literature review. Knowledge and Information Systems, 64(6). https://tinyurl.com/5n6kfn29