Abstract

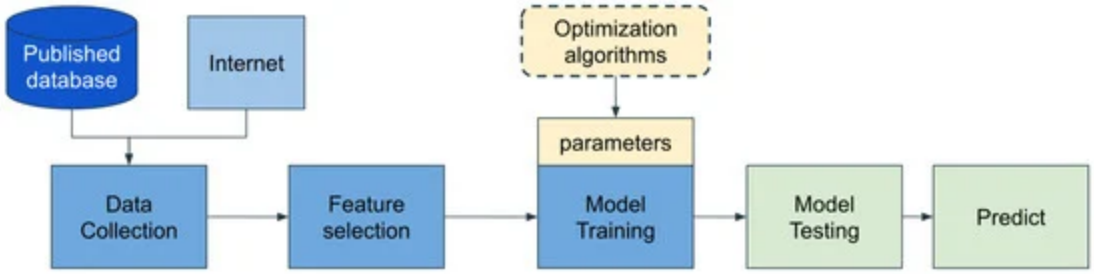
As the internet continues to evolve, the importance of network security becomes increasingly paramount. The internet's robust and accelerated growth relies on a secure online environment. An insecure online environment can lead to a decrease in the number of users due to a decline in users' trust. Since cyber-attacks are increasing with the evolution of the internet, it is essential for network security to evolve simultaneously. While there are numerous threats on the internet, one critical threat is phishing. Phishing is a form of cybercrime that involves tricking users into revealing personal information through deceptive links that lead to unauthorized access and financial fraud. The dynamic nature of network security, where defensive detection strategies are used against offensive phishing techniques, underscores the challenge of protecting users from phishing attacks. Traditional methods, like creating blacklists and whitelists, fall short in identifying novel phishing threats. For network security to evolve alongside attacks, adaptable approaches to predict and accurately identify emerging phishing sites must be implemented. This paper presents a phishing detection system using advancements in machine learning. It delves into examining existing anti-phishing strategies and thoroughly explores a machine-learning approach to the problem, including aspects of data gathering, feature extraction, model development, and performance evaluation.

Problem Statement

The prevalence and sophistication of phishing attacks have escalated with the growth of the internet, posing a significant threat to online security and user privacy. Traditional phishing detection methods, such as blacklists and whitelists, are increasingly ineffective against novel and evolving phishing techniques. There is a critical need for a more dynamic and adaptable approach to accurately identify and counteract these phishing threats. This project aims to develop and validate a machine learning based solution for phishing website detection, which can efficiently differentiate between legitimate and malicious websites. The solution should address the challenges of adapting to new phishing strategies, minimizing false positives, and ensuring timely detection to protect users from potential data and financial losses.

Background and Related Work

Phishing, a prevalent cyber threat, involves deceptive emails or messages that lure recipients to fraudulent websites to harvest their sensitive information like usernames, passwords, and credit card details for illicit financial gains. This process is illustrated in Figure 1, which displays the phishing attack lifecycle. Initially, attackers craft a fake website, mirroring a legitimate one. They often use tactics such as misspellings, similar characters, and other techniques to mimic the URL, particularly in the domain name and directory structure. For instance, the link "**https://aimazon.amz-z7acyuup9z0y16.xyz/v**" (accessed on 9 May 2021) is designed to imitate **https://www.amazon.com**. While browsers display the URL when hovering over links, the subtle differences often escape the average user's immediate recognition. Additionally, phishers replicate the look and feel of legitimate web pages by copying logos, layouts, and text, especially on pages requiring sensitive user input like login, payment, or password reset interfaces.



The subsequent phase involves distributing the phishing link through emails or other mediums like SMS, voice messages, QR codes, and fake apps [2]. With the rise of smartphones and social media, these dissemination channels have expanded. Attackers typically employ compelling language and imagery in these messages to entice users into clicking the links, often using tactics like impersonating customer service to create urgency or fear. Despite the random nature of these emails, a fraction of users, particularly those less security-savvy, fall prey to these tactics. Here social engineering plays a vital role, as attackers exploit the psychological manipulation to induce security lapses. Once the user clicks the link, they are directed to the fraudulent website on their device.

The third step involves collecting personal information from users on these counterfeit websites, which skillfully mimic real organizations' websites in appearance, including logos, names, and user interfaces. These websites primarily target information entry points such as login, password reset, and payment sections. When users input their sensitive data, it gets transmitted to servers controlled by the attackers.

The final stage sees attackers leveraging this stolen information. They may use it to access the victim's accounts on various websites, particularly when individuals reuse the same usernames and passwords. Some attackers further exploit this data for additional criminal activities. Since the first recorded instance of phishing in 1987 [6], the tactics have evolved with the internet's growth, such as targeting online payment platforms. According to the 2020 Internet Crime Report report, the Internet Crime Complaint Center (IC3) received 791, 790 cyberattack complaints and revealed that phishing constituted about 30% of these complaints. This underscores that phishing was the most complained type of cybercrime and caused more than 54 million USD in losses [1]. This underscores the prevalence of phishing and the importance for internet users to distinguish between genuine and counterfeit web pages.

In combating phishing, disrupting any stage of the attack can be effective. Below are some anti-phishing techniques starting from each stage.

1. **Web-Scraping:** Legitimate websites can deter attackers from copying their content by employing techniques like CSS sprites and replacing text with images. However, these methods have limitations. They can potentially degrade user experience by making the website less accessible, especially for users who rely on screen readers or other assistive technologies. Moreover, sophisticated attackers can circumvent these measures by using advanced scraping tools that can interpret and mimic human interaction with websites.
2. **Spam Filter Implementation:** Email services like Gmail, Yahoo, Outlook, and AOL use spam filters to block phishing emails. While traditional spam filters based on blacklists and whitelists have been effective to some extent, they struggle to keep up with the evolving tactics of phishers. Blacklist-based filters can't recognize new phishing sites that haven't been reported yet, leading to a lag in protection. Whitelists, on the contrary, can be too restrictive, potentially blocking legitimate emails. The reliance on predefined rules also makes these filters less adaptable to new phishing techniques [3]. These filters have evolved from basic blacklist/whitelist systems to

sophisticated machine learning models capable of identifying previously unknown spam. For example, Gmail can block about 100 million additional spam emails everyday with the implementation of a machine learning based spam filter [8].

3. **Fake Website Detection:** Many browsers now include security features to ward users of unsafe websites. Google Safe Browsing, for instance, uses a blacklist of known malicious URLs [4]. However, the use of blacklists of known malicious URLs are only as good as their most recent update. Attackers continually create new phishing sites, and there's often a delay before these sites are identified and added to blacklists. This delay opens up a window of vulnerability for users [10]. As attackers continually create new phishing sites, AI and machine learning models are increasingly vital in identifying these threats.
4. **Second Authorization Verification:** Additional verification steps, like facial recognition or voiceprint analysis, can prevent unauthorized account access even after attackers obtain user data. However, these verification steps also introduce complexity and potential privacy concerns. These methods may be inconvenient for users, leading to resistance in adoption. Moreover, biometric data, if compromised, cannot be easily changed like passwords, posing a significant security risk [7].

Machine learning offers a more robust approach to anti-phishing for several reasons. It offers adaptability as machine learning models can learn from new data and adapt to evolving phishing strategies. Unlike rule-based systems, they are not limited to predefined characteristics of phishing attacks and can detect novel threats [3]. Additionally, machine learning algorithms can automatically identify and extract relevant features from data, which is crucial in detecting sophisticated phishing attacks that might not be obvious to human analysts or traditional rule-based systems [9]. Machine learning models can also process large volumes of data quickly and efficiently, making them suitable for real-time phishing detection across vast networks and systems [6].

While traditional anti-phishing strategies provide some level of protection, their effectiveness is often limited by their static nature and inability to adapt to new and evolving threats. Machine learning offers a more dynamic and effective approach by continuously learning from new data and patterns, thereby enhancing the detection and prevention of phishing attacks.

Security Analysis/Threat Model

1. **Threat Identification**
    a. *Data Poisoning Attacks:* Adversaries may attempt to manipulate the training data by injecting malicious inputs, leading to a corrupted model that fails to identify phishing attempts correctly.
    b. *Evasion Techniques:* Attackers may employ sophisticated evasion tactics to create phishing content that bypasses detection, such as using advanced obscuring techniques or leveraging AI-generated content that mimics legitimate communications.
    c. *Model Stealing or Inversion:* Attackers may attempt to steal the machine learning model to understand its inner workings and develop strategies to circumvent it, or use model inversion attacks to uncover sensitive information about the training data.
    d. *Adversarial Machine Learning:* Attackers can exploit the model by understanding its decision boundaries and crafting inputs that are classified incorrectly.
2. **Vulnerability Analysis**

a. *Overfitting:* The model may overfit to the training data, reducing its ability to generalize to novel phishing attacks.
b. *Underfitting:* If the model is too simple, it might fail to capture the complexity of phishing attacks, leading to high false negative rates.
c. *Data Quality and Availability:* The efficacy of the model heavily relies on the quality and comprehensiveness of the training data. Limited or biased data can hinder the model's performance.
d. *Model Interpretability:* Complex models like deep neural networks may act as black boxes, making it difficult to understand why certain decisions are made, which is crucial for continuous improvement and trust in the system.

3. **Attack Surface Evaluation**
    a. *APIs and Interfaces:* The interfaces through which the model interacts with other systems can be vulnerable to attacks, including injection attacks or unauthorized access.
    b. *Update Mechanisms:* The process of updating the model or its data sources might expose the system to risks, especially if updates are not securely managed.
    c. *Integration Points:* The system's integration with email servers, web browsers, and other platforms can be potential points of exploitation.
4. **Risk Mitigation Strategies**
    a. *Regular Model Retraining:* To counter data poisoning and evasion techniques, regularly retrain the model with new and diverse datasets.
    b. *Robustness Testing:* Employ adversarial testing to evaluate the model's resilience against deliberately crafted malicious inputs.
    c. *Secure Model Serving:* Implement robust security measures for APIs and model serving infrastructure to prevent unauthorized access and tampering.
    d. *Transparency and Interpretability:* Utilize models or tools that offer interpretability to understand model decisions, which aids in identifying potential biases or weaknesses.
    e. *Continuous Monitoring:* Implement continuous monitoring systems to detect and respond to anomalies in real-time, indicating potential evasion or poisoning attempts.
    f. *Data Privacy and Security:* Ensure that data collection, storage, and processing comply with privacy standards and are secure against unauthorized access.
5. **Compliance and Ethical Considerations**
    a. *Privacy Regulations Compliance:* Ensure compliance with data protection regulations like GDRP, especially when dealing with user data.
    b. *Ethical Use of Data:* Establish guidelines for the ethical use of data, particularly when user-generated data for training the model.
6. **Incident Response Plan**
    a. Develop a comprehensive incident response plan that includes procedures for handling security breaches, model failures, or data leaks. This plan should also include mechanisms for quickly updating the model in response to new threats.

Description/Technical Analysis of Solution

We used two different models to develop our phishing detection system. One of these models was, XGBoost (eXtreme Gradient Boosting), which is a highly efficient and flexible algorithm for

supervised machine learning tasks. It is an implementation of gradient-boosted decision trees designed for speed and performance. Gradient boosting is a technique where new models are added to correct the errors made by existing models. Models are sequentially added until no further improvements can be made. XGBoost can handle large, complex datasets and includes a variety of techniques for regularization, tree pruning, and handling missing values, which prevent overfitting and make it more robust than regular gradient boosting. Parallel construction of decision trees across all CPU cores during the training phase significantly speeds up computation. XGBoost also has a built-in routine to run cross-validation at each iteration of the boosting process. This helps in determining the exact model configuration that makes the best use of the training data. There is also a built-in routine to handle missing data. When a missing value is encountered at a split, it tries both directions to see which path gives the best results.

The other model that was trained and tested was a random forest classifier. This is a powerful and versatile machine learning model that we used for the classification of phishing websites. It is a type of ensemble learning method, where multiple decision trees are combined to improve overall performance. Each tree in a random forest gives a prediction and the final output in classification is determined by the majority vote of all trees. Unlike a single decision tree, which might be prone to overfitting, a random forest model averages out biases and variances. This leads to better generalization of novel data. Random forest classification introduces randomness by selecting a random subset of features at each split in the decision tree. This leads to diversity among trees and reduces the correlation between them, which decreases the model's variance.

The accuracy of our trained XGBoost model's performance on the test data was quite high as it was computed to be 0.861. The precision score was 0.961, recall was 0.748, F1 score was 0.841, and the AUC-ROC was 0.859. The accuracy score of our trained random forest model's performance on the testing and training data was lower than the XGBoost model. The accuracy for the random forest model was 0.781. The precision score was 0.781, the recall score was 0.564, the F1 score was 0.718, and the AUC-ROC was 0.779. Moreover, for both models, the accuracies on the training and testing datasets were very similar, indicative of the model's ability to generalize well to novel data. This implies that the model has learned the underlying patterns in the data rather than memorizing the training set. The accuracy scores also indicate that the model is not overfitting or underfitting. Furthermore, similar test and train accuracies suggests that the test set is representative of the overall data distribution and the scenarios in which the model will be used. However, the results are a bit unusual since the accuracy score on the training data appeared to be lower than that on the testing data. Since the difference is very minor it can be due to a result of randomization as the web traffic feature is constantly changing and the list of phishing URLs on PhishTank is updated hourly.

Methodology and Setup

1. Collect datasets containing phishing and legitimate websites from open source platforms.
    a. Phishing URLs: https://www.phishtank.com/developer_info.php
        i. This site is updated hourly.
    b. Legitimate URLs: https://www.unb.ca/cic/datasets/url-2016.html
2. Conduct feature extraction from the URL database.
    a. Read the phishing and legitimate link datasets using the Python Pandas library that can be installed with the command pip install pandas.

    b. Collect 5,000 phishing URLs and 5,000 legitimate URLs randomly.
    c. Address bar-based features to extract (Python packages: urllib.parse, ipaddress, re)
        i. Domain of URL
        ii. IP Address in URL
        iii. "@" Symbol in URL
        iv. Length of URL
        v. Depth of URL
        vi. Redirection "//" in URL
        vii. "http/https" in Domain name
        viii. Using URL Shortening Services "TinyURL"
        ix. Prefix or Suffix "-" in Domain
    d. Domain based features to extract (Python packages: re, bd4, whois, urllib, datetime)
        i. DNS Record
        ii. Website Traffic (using checkpagerank.net)
        iii. Age of Domain
        iv. End Period of Domain
    e. HTML and JavaScript based features (Python package: requests)
        i. IFrame Redirection
        ii. Status Bar Customization
        iii. Disabling Right Click
        iv. Website Forwarding
3. Compute URL features.
    a. Store all the features in a list.
4. Analyze and preprocess the dataset by using exploratory data analysis (EDA) techniques.
    a. Create histograms to display how the data is distributed and how the features are related to each other.
    b. Use the pandas describe method to obtain statistics regarding the data.
        i. Domain column is dropped as it does not have any significance in training the model. Leaves us with 16 features and a target column.
        ii. Check for null or missing values.
5. Shuffle and divide the dataset into training and testing sets.
    a. Use the train_test_split function from the scikit-learn model_selection module.
6. Run XGBoost classifier on the dataset.
    a. There is a Python xgbost package that can be installed with the command pip install xgboost. Import XGBClassifier and initiate the model with a learning rate of 0.4 and max depth of 7.
    b. Fit the model on the training data.
    c. Predict the target value from the model for the test and train samples.
7. Repeat step 6 but using the RandomForestClassifier from scikit-learn.
8. Display the evaluation result considering accuracy metrics.
    a. Import accuracy_score from scikit-learn metrics module.
    b. Compute the accuracy score of the model on the training and testing data.

Conclusion/Future Work

In this study, we developed a phishing detection model using the XGBoost algorithm, which is known for its efficiency and effectiveness in handling complex datasets. Our model demonstrates promising results in identifying phishing attempts, leveraging XGBoost's advanced capabilities in handling imbalanced data, its robustness to overfitting, and its ability to process large datasets swiftly.

The XGBoost model achieved high accuracy, which indicates its strong potential in the field of cybersecurity, specifically phishing detection. The model highlighted the significance of certain features in phishing detection, offering insights into the tactics used by phishers and aiding in the development of more targeted countermeasures. Moreover, the model's efficiency in training and prediction makes it suitable for real-time phishing detection, a crucial factor in cybersecurity. The model's performance on novel data was also commendable, proving that it has decent generalization capabilities.

To further enhance the phishing detection model and address current limitations, the following future work is proposed. The incorporation of a more diverse and extensive dataset, including the latest phishing examples, will help improve the model's robustness and adaptability to evolving phishing tactics. Experimenting with additional features, like website content analysis and user behavior metrics, could provide deeper insights and improve detection accuracy. Combining XGBoost with other machine learning algorithms in an ensemble method could further enhance the model's performance by leveraging the strengths of various algorithms. Deploying the model in a real-world environment and conducting extensive testing will provide practical insights into its effectiveness and areas for improvement. Investigating and developing strategies to counteract advanced evasion techniques used by sophisticated phishers, such as AI-generated content and dynamic websites, could improve phishing detection. An automated retraining pipeline can also be developed to allow the model to continuously learn from novel data, ensuring it remains effective against new phishing techniques.

References

[1] 2020 Internet Crime Report. Federal Bureau of Investigation. Available online: **https://www.ic3.gov/Media/PDF/AnnualReport/2020_IC3Report.pdf** (accessed on 21 March 2021).

[2] Alsariera, Y.A.; Adeyemo, V.E.; Balogun, A.O.; Alazzawi, A.K. AI Meta-Learners and Extra-Trees Algorithm for the Detection of Phishing Websites. *IEEE Access* **2020**, *8*, 142532–142542. [**Google Scholar**] [**CrossRef**]

[3] Basit, A.; Zafar, M.; Liu, X.; Javed, A.R.; Jalil, Z.; Kifayat, K. A comprehensive survey of AI-enabled phishing attacks detection techniques. *Telecommun. Syst.* **2020**, *76*, 139–154. [**Google Scholar**] [**CrossRef**] [**PubMed**]

[4] Google Safe Browsing. Google.com. 2014. Available online: **https://safebrowsing.google.com/** (accessed on 18 July 2021).

[5] Jain, A.K.; Gupta, B.B. A novel approach to protect against phishing attacks at client side using auto-updated white-list. *EURASIP J. Inf. Secur.* **2016**. [**Google Scholar**] [**CrossRef**][**Green Version**]

[6] Jerry, F.; Chris, H. System Security: A Hacker's Perspective. In Proceedings of the 1987 North American conference of Hewlett-Packard business computer users, Las Vegas, NV, USA, 20–25 September 1987. [**Google Scholar**]

[7] Kalaharsha, P.; Mehtre, B.M. Detecting Phishing Sites—An Overview. *arXiv* **2021**, arXiv:2103.12739. [**Google Scholar**]

[8] Kumaran, N. Spam Does Not Bring Us Joy—Ridding Gmail of 100 Million More Spam Messages with TensorFlow. Google Cloud Blog. Available online: **https://cloud.google.com/blog/products/g-suite/ridding-gmail-of-100-million-more-spam-messages-with-tensorflow** (accessed on 6 February 2019).

[9] Singh, C. Phishing Website Detection Based on Machine Learning: A Survey. In Proceedings of the 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 6–7 March 2020. [**CrossRef**]

[10] Vijayalakshmi, M.; Shalinie, S.M.; Yang, M.H. Web phishing detection techniques: A survey on the state-of-the-art, taxonomy and future directions. *IET Netw.* **2020**, *9*, 235–246. [**Google Scholar**] [**CrossRef**]