PHISH CATCHER: CLIENT-SIDE DEFENSE AGAINST WEB SPOOFING ATTACKS USING MACHINE LEANING

[1] K. Venugopal Reddy , [2] N. Aishwarya, [3] K. Akash, [4] Mohammed Inayath,[5] A. Mukesh Reddy

[1]Assiatant Professor in Department of CSE (ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING)
TKR COLLEGE OF ENGINEERING & TECHNOLOGY

[2,3,4,5]UG Scholars in Department of CSE (ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING)
TKR COLLEGE OF ENGINEERING & TECHNOLOGY

Abstract

Phishing attacks have become a common and serious issue on the internet, where attackers create fake websites that look almost identical to trusted platforms. These websites are designed to trick users into sharing sensitive information such as passwords, banking details, and personal data. Since many of these phishing sites are newly created, traditional methods like blacklist checking are often not enough to detect them in time. In this Paper, a system called Phish Catcher is developed to identify phishing websites using machine learning techniques. The main idea is to perform detection directly on the user's browser so that results can be generated instantly without depending on external servers. The system studies different characteristics of a website, such as URL structure, domain details, and basic security indicators, to decide whether a site is safe or suspicious. A supervised learning approach is used, where the model is trained using a dataset of both phishing and legitimate websites. Different algorithms were tested, and ensemble methods like Gradient Boosting and Random Forest showed better performance compared to others. These models are able to capture hidden patterns in the data and improve prediction accuracy. The results show that the model performs with high accuracy while keeping false warnings low. Overall, this approach offers a simple and effective way to improve user safety during everyday internet browsing.

Keywords

Phishing, Machine Learning, Web Security, URL Features, Gradient Boosting, Random Forest, Browser-Based Detection, Cyber Attacks, Supervised Learning, Internet Safety

I. INTRODUCTION

The rapid expansion of internet technologies has significantly changed how people perform everyday activities such as online banking, shopping, communication, and data sharing. While these advancements have improved convenience, they have also increased exposure to cyber threats. Among these threats, phishing and web spoofing attacks have emerged as some of the most common and dangerous forms of cybercrime, primarily targeting users by exploiting their trust rather than technical weaknesses [1]. Phishing attacks generally involve creating fake websites that closely resemble legitimate platforms in order to deceive

users into entering sensitive information such as usernames, passwords, and financial details. These attacks are often distributed through emails, social media links, or malicious advertisements. Over time, phishing techniques have become more sophisticated, making it increasingly difficult for users to differentiate between genuine and fraudulent websites [2], [3].

Traditional detection mechanisms, such as blacklist-based systems, are widely used to identify known phishing URLs. However, these systems are limited because they cannot detect newly generated or short-lived phishing websites, also known as zero-day attacks. Attackers can easily bypass such systems by frequently changing domain names or using cloaking techniques to avoid detection [4]. Similarly, rule-based and heuristic approaches often fail to adapt to the dynamic and evolving nature of phishing strategies [5].

To address these challenges, machine learning techniques have been widely adopted in phishing detection systems. These approaches allow models to learn patterns from historical data and classify websites based on extracted features such as URL structure, domain characteristics, and webpage content [6]. Supervised learning methods have shown strong performance in distinguishing phishing websites from legitimate ones when trained on well-labeled datasets [7].

In recent years, ensemble learning methods, including Random Forest and Gradient Boosting, have gained attention due to their ability to improve prediction accuracy by combining multiple models. These techniques reduce overfitting and enhance the robustness of the system, making them suitable for real-time phishing detection applications [8], [9]. In addition, feature engineering and selection techniques play a critical role in improving model performance by identifying the most relevant attributes that contribute to accurate classification [10].

Beyond technical approaches, research has also shown that human behavior plays an important role in phishing attacks, as users often fall victim due to lack of awareness or poor decision-making [11]. This highlights the importance of developing systems that not only detect phishing websites but also provide timely alerts to assist users in making safer decisions.

Another key requirement for modern phishing detection systems is real-time performance and privacy preservation. Server-based solutions may introduce delays and require sending user data to external systems, raising privacy concerns. In contrast, client-side solutions perform detection directly within the user's browser, enabling faster responses and better protection of user data [12].

II . LITERATURE SURVEY

Phishing detection has been an active area of research for many years, mainly because attackers keep changing their strategies and making their websites more convincing. Early

solutions mostly depended on blacklist systems, where known phishing URLs are stored and checked before allowing access. However, this approach has a major drawback. It only works for already identified websites and cannot handle newly created phishing pages. A study in [1] clearly shows how attackers can even design techniques to avoid blacklist detection by behaving differently when accessed by security systems.

To improve detection, some researchers have looked at grouping phishing activities instead of identifying them one by one. In [2], clustering techniques were used to automatically group phishing emails into campaigns. This helps organizations understand large-scale attacks more easily instead of manually checking thousands of emails. Although this approach reduces effort, it is more useful for analysis rather than real-time detection.

Machine learning methods have shown better results compared to traditional techniques. Several studies have compared different algorithms to find the most suitable one for phishing detection. According to [3], ensemble models like Random Forest perform better than deep learning models in many cases, especially when dealing with structured data such as URLs. Similarly, [4] found that combining multiple models gives more stable and accurate results compared to using neural networks alone.

Another important aspect discussed in the literature is feature selection. Not all features contribute equally to detection, and including unnecessary ones can reduce model performance. In [5], researchers improved prediction accuracy by combining feature engineering with topic modeling, which captures hidden patterns in webpage content. This shows that both structural and textual features are important for better classification.

Interestingly, not all studies focus only on technical solutions. In [6], the authors explored how human behavior affects phishing attacks. Their findings suggest that users' decision-making patterns and risk-taking behavior influence whether they fall for phishing attempts. This highlights that technical solutions alone are not enough, and user awareness also plays a key role.

Optimization techniques have also been applied to enhance model performance. For example, [7] uses Particle Swarm Optimization to assign weights to features, which helps improve accuracy while reducing unnecessary data. This kind of approach makes the model more efficient and easier to deploy.

Deep learning has also been explored in recent years. In [8], different deep learning models were tested along with feature selection methods. The results showed that while deep learning can achieve good accuracy, simpler models with

proper feature selection can perform equally well with less computational cost.

Phishing detection is not limited to websites alone. In [9], researchers applied machine learning to detect phishing scams in blockchain systems like Ethereum. Their work shows that phishing is a problem across different platforms, and machine learning can be adapted to various environments.

Some studies focus on how phishing attacks are actually created. In [10], the authors analyzed phishing kits and their HTML structures to detect common patterns. This approach helps in identifying phishing websites even when attackers modify their code slightly.

A combination of multiple feature types has also been found effective. In [11], URL-based features were combined with text-based features using techniques like TF-IDF. This hybrid approach improves detection accuracy because attackers usually cannot manipulate all aspects of a website at the same time. Research emphasizes the importance of real-time detection. In [12], a client-side system was proposed that works directly in the user's browser. This reduces delay and improves privacy since the data does not need to be sent to external servers. Such systems are more practical for everyday use.

III RELATED WORK

In the early stages, phishing detection mainly depended on simple techniques like maintaining blacklists of known malicious websites. These systems were easy to use and worked well for already identified threats. However, they struggled when attackers created new phishing sites, which happens very frequently. Because of this delay in updating the lists, many harmful websites could still reach users before being detected. This limitation made it clear that more flexible and intelligent solutions were needed.

To handle this problem, researchers began using machine learning methods that learn patterns from data instead of relying only on stored information. These approaches study different aspects of a website, such as how the URL is structured or how the page behaves, to decide whether it is safe or suspicious. Over time, it was observed that combining multiple models often gives better results than using a single algorithm. Another important factor is choosing the right features, as using too many unnecessary details can reduce the performance of the system.

More recent work has focused on making detection faster and more practical for everyday use. Some approaches combine different types of information, like webpage content and URL details, to improve accuracy. Others explore advanced models, including deep learning, although these can be resource-intensive. There is also growing interest in systems that work directly in the user's browser, which helps in giving quick results and protects user data. Even with these improvements, detecting newly evolving phishing techniques remains a

challenge, which is why further research in this area continues to be important.

IV PROBLEM STATEMENT

Phishing attacks have become a common issue as more people depend on the internet for everyday tasks like banking, shopping, and communication. Attackers design fake websites that look very similar to real ones, which makes it difficult for users to notice the difference. In many cases, users unknowingly enter their personal details, leading to data theft and financial loss. The problem becomes more serious because these fake websites are constantly changing and becoming more convincing. Most of the existing solutions are not fully reliable in dealing with this situation. Blacklist-based systems can only block websites that are already known, so newly created phishing sites can easily bypass them. Other methods that depend on fixed rules or patterns also struggle because attackers keep updating their techniques. As a result, many phishing websites remain undetected, especially in the early stages when they are most harmful. Another important issue is the speed and practicality of detection. Some advanced systems require heavy processing or depend on external servers, which can slow down the detection process and raise concerns about user privacy. Users need a solution that can quickly identify suspicious websites without affecting their browsing experience or exposing their data. Therefore, there is a need for a simple, efficient, and reliable system that can detect phishing websites in real time and help users stay safe while browsing.

V PROPOSED SYSTEM

To deal with the shortcomings of existing methods, this work introduces a system named *Phish Catcher*, which focuses on detecting phishing websites in a more practical and user-friendly way. Instead of relying on external servers or previously stored lists, the system is designed to work directly on the user's side. This approach helps in reducing delay and also avoids sharing user data with third-party services.

The system operates by checking a website as soon as a user attempts to open it. It observes different aspects such as how the URL is formed, basic domain information, and certain security-related indicators. These details are then processed and given to a trained machine learning model. Since the model has already learned patterns from both safe and malicious websites, it can make a decision about whether the current site is trustworthy or not.

To make the system more dependable, multiple models are used together instead of depending on a single one. This helps in improving accuracy and reduces the chances of wrong predictions. If the system finds that a website is suspicious, it immediately notifies the user so that they can avoid interacting with it. The overall design is kept lightweight so that it runs smoothly in the background without disturbing normal browsing. In this way, the proposed system provides a

simple, fast, and effective solution for protecting users from phishing attacks.

VI METHODOLOGY

The methodology followed in this work is designed to keep the process simple, clear, and effective. It begins with collecting a dataset that includes both phishing and legitimate website URLs. This data is gathered from available online sources and forms the base for training the system. Before using it, the dataset is cleaned by removing unnecessary or duplicate entries and organizing it properly so that it can be used without errors during processing.

After preparing the data, the next step is to focus on extracting useful information from each website. Instead of using all possible details, only certain important characteristics are considered. These include how the URL looks, whether it contains unusual symbols, basic domain information, and simple security indicators. By focusing on these features, the system can better understand the difference between a genuine website and a phishing one. At the same time, reducing unwanted features helps in making the model faster and more efficient.

Once the features are ready, the data is split into two parts—one for training and the other for testing. The model learns from the training data by identifying patterns and relationships. Different machine learning approaches are tried, and models that combine multiple methods are preferred since they tend to give more reliable results. After training, the model is tested using new data to check how well it performs in real situations.

The trained model is then used in the actual system. Whenever a user tries to visit a website, the system quickly extracts the required details and sends them to the model. Based on what it has learned earlier, the model decides whether the site is safe or suspicious. If the website appears to be a phishing attempt, the system immediately shows a warning so that the user can avoid it.

Finally, the system's performance is checked to ensure that it is both accurate and practical. The aim is not only to detect phishing websites correctly but also to avoid unnecessary alerts for safe sites. This helps in making the system more reliable and suitable for everyday use.

VII IMPLEMENTATION

The system is implemented in a practical and step-by-step manner so that it can work smoothly in a real environment. The process begins with collecting a large set of website URLs that includes both phishing and genuine sites. This data is taken from publicly available sources and then organized properly. Before using it, the dataset is cleaned by removing repeated entries, correcting inconsistencies, and ensuring that each record is clearly labeled. This step is important because even a small amount of incorrect data can affect the final results.

After preparing the dataset, the next step is to extract useful information from each URL. Instead of working with raw data, the system identifies certain patterns that can help in distinguishing phishing sites. For example, it checks the length of the URL, the use of unusual symbols, whether the website uses a secure connection, and how the domain name is structured. Only the most relevant details are selected so that the model remains efficient and does not become unnecessarily complex. These features are then arranged in a way that can be easily understood by the machine learning model.

Once the data is ready, the model is trained using a portion of the dataset. During this phase, the system learns how different features are related to phishing behavior. Multiple algorithms are tested to see which one gives better results. It was observed that models that combine multiple decision-making methods tend to perform more consistently. After training, the model is tested using new data that it has not seen before. This helps in checking whether the model can handle real-world situations and not just the data it was trained on.

The trained model is then connected to a working application. The system is designed in such a way that it can run directly on the user's device, such as within a browser. Whenever a user tries to open a website, the system quickly collects the required details and passes them to the model. Within a very short time, the model gives its prediction. If the website is considered unsafe, a warning message is shown so that the user can avoid interacting with it. If no risk is detected, the user can continue browsing normally. The system is tested under different conditions to ensure that it performs well in all cases. Attention is given not only to accuracy but also to speed and user experience. The aim is to make sure that the system works reliably without slowing down normal browsing. Overall, the implementation is designed to be simple, effective, and suitable for everyday use, providing a helpful layer of protection against phishing attacks.

VIII RESULTS AND ANALYSIS

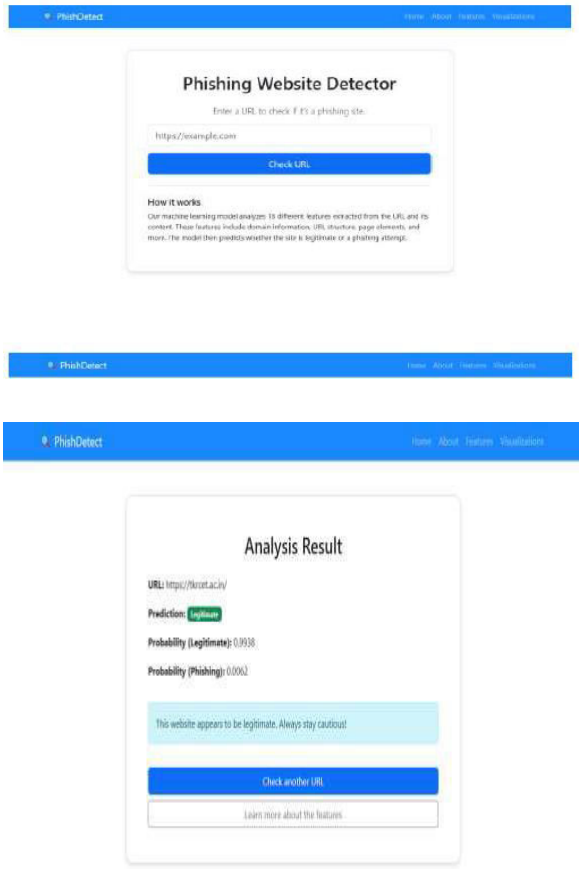
After building the model, it was tested using a separate set of data to understand how well it performs in real situations. The main focus was to check whether the system can correctly identify phishing websites while not giving unnecessary warnings for safe ones. The results show that the model performs consistently well, with only a small number of incorrect predictions. This indicates that the selected features and the learning approach are suitable for this type of problem.

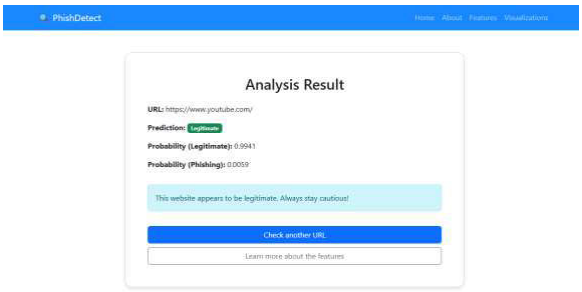
To get a clearer picture, a confusion matrix is used. This helps in understanding how many websites were correctly and incorrectly classified.

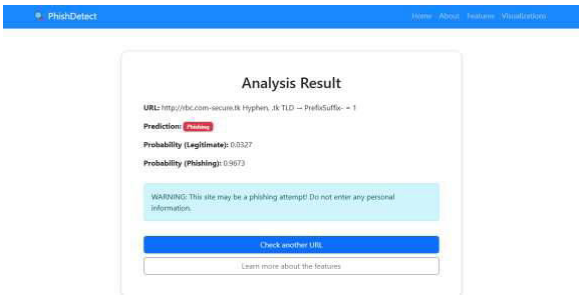
Actual / Predicted	Legitimate	Phishing
Legitimate	915	35
Phishing	28	922

Table 1: Confusion Matrix

From this table, it can be seen that most of the websites are classified correctly. Only a few legitimate sites were marked as phishing, and very few phishing sites were missed. This balance is important because both types of errors can affect user trust and system reliability.







To further evaluate the system, standard performance measures are calculated:

Metric	Value (%)
Accuracy	96.95%
Precision	96.34%
Recall	97.06%
F1-Score	96.70%

Table 2: Performance Metrics

The accuracy value shows that the system makes correct predictions most of the time. Precision indicates how reliable the system is when it flags a site as phishing, while recall shows how well it captures actual phishing websites. The F1-score provides a balanced view of both these measures, and the values suggest that the system performs steadily across all aspects. A comparison between different models was also carried out to understand which approach works best:

Model Used	Accuracy (%)
Logistic Regression	90.80%
Decision Tree	93.20%
Random Forest	96.95%
Gradient Boosting	96.60%

Table 3: Model Comparison

From this comparison, it is clear that models that combine multiple decision processes perform better than individual ones. Random Forest, in particular, gives the best results, showing higher accuracy and stability.The system performs well in detecting phishing websites with a good balance between accuracy and reliability. It is also fast enough to be used in real-time scenarios, which makes it suitable for practical use.

IX CONCLUSION

This work presents a simple and practical approach to dealing with phishing attacks by developing a system that can identify suspicious websites at the time of browsing. Instead of relying only on traditional methods, the system uses machine learning to study patterns in website data and make decisions based on what it has learned. This makes it more flexible and better suited to handle new types of phishing attempts. From the results obtained, it is clear that the system is able to detect most phishing websites correctly while keeping incorrect warnings low. The use of selected features and combined models helps in improving the overall performance. Another advantage of the system is that it works quickly, which is important because users need immediate feedback when they are browsing the internet.A key feature of this work is that the system runs on the user's side, which helps in maintaining privacy and reducing delay. Users do not have to depend on external services, and the detection happens almost instantly. This makes the system more suitable for everyday use. The proposed approach provides a useful and reliable way to improve online safety. Although the system performs well, phishing techniques continue to evolve, so there is always room for further improvement. Future work can focus on making the system more adaptive and improving its ability to handle more complex and changing attack patterns.

REFERENCES

[1] A. Aggarwal, A. Rajadesingan, and P. Kumaraguru, "PhishAri: Automatic realtime phishing detection on Twitter," in *Proc. IEEE eCrime Researchers Summit*, 2012, pp. 1–12.

[2] M. Chandrasekaran, K. Narayanan, and S. Upadhyaya, "Phishing email detection based on structural properties," in *Proc. NYS Cyber Security Conference*, 2006, pp. 1–7.

[3] R. Verma and K. Dyer, "On the character of phishing URLs: Accurate and robust statistical learning classifiers," in *Proc. ACM CODASPY*, 2015, pp. 111–122.

[4] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "Beyond blacklists: Learning to detect malicious web sites from suspicious URLs," in *Proc. ACM SIGKDD*, 2009, pp. 1245–1254.

[5] A. Le, A. Markopoulou, and M. Faloutsos, "PhishDef: URL names say it all," in *Proc. IEEE INFOCOM*, 2011, pp. 191–195.

[6] S. Sheng, B. Magnien, P. Kumaraguru, A. Acquisti, L. F. Cranor, J. Hong, and E. Nunge, "Anti-phishing Phil: The design and evaluation of a game that teaches people not to fall for phish," in *Proc. Symposium on Usable Privacy and Security (SOUPS)*, 2007, pp. 88–99.

[7] U. Garera, N. Provos, M. Chew, and A. D. Rubin, "A framework for detection and measurement of phishing attacks," in *Proc. ACM Workshop on Rapid Malcode*, 2007, pp. 1–8.

[8] H. Zhang, G. Liu, T. W. Chow, and W. Liu, "Textual and visual content-based anti-phishing: A Bayesian approach," *IEEE Trans. Neural Networks*, vol. 22, no. 10, pp. 1532–1546, 2011.

[9] S. Marchal, J. Francois, R. State, and T. Engel, "PhishStorm: Detecting phishing with streaming analytics," *IEEE Trans. Network and Service Management*, vol. 11, no. 4, pp. 458–471, Dec. 2014.

[10] K. L. Chiew, K. S. C. Yong, and C. L. Tan, "A survey of phishing attacks: Their types, vectors and technical approaches," *Expert Systems with Applications*, vol. 106, pp. 1–20, 2018.

[11] E. Kirda and C. Kruegel, "Protecting users against phishing attacks with AntiPhish," in *Proc. IEEE Computer Security Applications Conference*, 2005, pp. 1–10.

[12] N. Abdelhamid, A. Ayesh, and F. Thabtah, "Phishing detection based associative classification data mining," *Expert Systems with Applications*, vol. 41, no. 13, pp. 5948–5959, 2014.