iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: https://doi.org/10.22214/ijraset.2026.79258

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

Phishing Website Detection Using Machine Learning

Saud T. Ali, Tatheer Hussain, Tooba Wahab, Sifatullah Siddiqui
Department of Computer Science, Integral University Lucknow, India – 226026

Abstract: ***Cyber deception in the form of phishing website creation has been identified as one of the most critical challenges in the field of cybersecurity in the contemporary digital era. Phishing is a form of social engineering attacks that tricks users into divulging their valuable login details, bank account information, and identity details by masquerading as genuine online platforms. With the increasing cunningness of attackers and the exponentially growing number of internet users across the globe, rule-based and static blacklists have been proven to be grossly inefficient. This paper presents a supervised machine learning approach for phishing website detection through automated feature learning, such as URL lexicography, graph-based features, and domain-based features. A thoughtfully designed dataset comprising both phishing and genuine URLs is developed, preprocessed, and presented to a range of classification algorithms, such as Random Forest, Support Vector Machine (SVM), and Logistic Regression. The results clearly indicate that ensemble learning models, specifically Random Forest, are substantially more accurate than rule-based models. This paper also touches upon the real-world complexities of adversarial attacks, class imbalance, and the challenge of designing generalizable feature spaces and presents a roadmap for future research on developing adaptive anti-phishing systems.***

Keywords: Phishing Detection, Machine Learning, Cybersecurity, Random Forest, Feature Extraction, URL Analysis.

I. INTRODUCTION

As the role of digital infrastructure in commerce, communication, and governance becomes increasingly central, the corresponding threat environment has become increasingly complex. Cyber attackers now use more than just technical vulnerabilities; increasingly, they target the weakest part of any system—the human element of the user. Social engineering attacks, which target the user psychologically rather than exploiting software vulnerabilities, have become the most common mode of intrusion. In this area, phishing attacks occupy a disturbing niche: by displaying spoofed websites that are visually indistinguishable from the real thing, attackers have been able to harvest login credentials, financial, and identity information with alarming success. Software such as HTTrack has greatly reduced the technical expertise required to create a working replica of a website, making it virtually cost-free for would-be attackers.

The economic cost of phishing attacks is high and has been extensively documented. Statistics compiled by the FBI Internet Crime Complaint Center (IC3) show that in 2018 alone, internet-facilitated fraud resulted in losses of over $2.7 billion, with business email compromise attacks alone accounting for over $1.2 billion of that amount [4]. Projections included in the Microsoft Computing Safer Index Report suggest that the annual global cost of phishing attacks may reach as high as $5 billion [5]. These statistics point to a serious problem: while security experts may be able to spot phishing attacks with careful analysis, the average user simply does not have the training or the time to do so, leaving them vulnerable to attack and requiring automated solutions to address.

For nearly two decades, the primary means of defending against phishing attacks has relied on blacklists—pre-crafted lists of known malicious URLs and IP addresses maintained by security vendors. While easy to implement, this model has a critical flaw in its architecture—it is purely reactive. New phishing domains are registered and decommissioned at a pace that far outstrips manual cataloguing. Attackers further exploit this lag through techniques such as fast-flux DNS rotation, which continuously cycles the IP addresses associated with a malicious hostname, and through algorithmically generated domain names that evade pattern-based filters. The consequence is a persistent blind spot for so-called zero-hour attacks — phishing campaigns that inflict damage in the window before any detection signature exists. Heuristic approaches attempt to bridge this gap by codifying known structural properties of phishing pages, but they introduce their own drawbacks, most notably an elevated false positive rate that disrupts legitimate user activity.

Machine learning offers a structurally different — and more promising — path. Rather than operating from a fixed enumeration of known threats, ML models derive classification rules directly from data, enabling them to generalise to novel attack patterns not present in any prior catalogue.

This study benchmarks a set of widely used supervised classifiers, namely SVM, Naive Bayes, Decision Trees, Random Forest, and Artificial Neural Networks, against a standardised phishing URL dataset, with the goal of identifying which algorithmic approaches best balance accuracy, training efficiency, and real-world deployability.

The paper proceeds as follows. Section II reviews the existing literature on ML-based phishing detection. Section III details the data collection process and feature engineering methodology. Section IV reports experimental results across all evaluated classifiers. Section V presents conclusions, and Section VI outlines future development directions.

II. RELATED WORK

A considerable amount of literature has been devoted to the study of phishing URL detection using machine learning, with authors exploring a wide range of feature sets and learning strategies. One of the early works by Kiruthiga and Akila [1] proved the feasibility of automatic URL analysis for phishing URL detection. Their system used 16 carefully designed features covering the age of domain registration, statistical properties of URLs at the character level, and the presence of anomalous prefixes or suffixes, extracted from a unified dataset sourced from the UCI Machine Learning Repository and PhishTank. When tested on the feature set using SVM, Naive Bayes, and Random Forest, the ensemble learning strategy showed relatively better robustness to both false positives and false negatives, with an overall accuracy of 92.67%. Notably, the work emphasized the superior flexibility of model-based solutions over blacklisting, as the model-based solution could mark potentially phishing URLs based on their structural anomalies, even if they were not pre-cataloged as such.

A related study by Kulkarni and Brown [2] explored a larger feature set of 30 attributes, including not only URL-level features but also HTML layout and JavaScript execution features. The study, conducted in the R programming environment and using data from the UCI repository, represented all features using ternary values (−1, 0, 1) to normalize input features before applying Decision Tree and SVM classifiers. The SVM classifier performed better with 92.13% accuracy, as opposed to 91.13% for the decision tree classifier. The authors attributed the performance gain to the SVM classifier's capability to define non-linear decision boundaries using kernel functions, which helped when the feature space was such that phishing and benign URLs were not linearly separable.

A broader comparative study by Shahrivari et al. [3] systematically evaluated seven algorithms — Logistic Regression, Decision Trees, Random Forest, SVM, K-Nearest Neighbours, and Artificial Neural Networks — within a unified experimental framework. Their central finding reinforced a recurring theme in the literature: ensemble and kernel methods consistently outperform simpler linear classifiers, and the margin of superiority widens as the feature space becomes richer and more complex. The study also drew attention to the arms-race dynamic inherent in phishing detection — adversaries who study detection systems can engineer URLs to evade specific features — and argued that adaptive, continuously retrained models represent the most durable long-term defence posture.

III. METHODOLOGY

A. Dataset Construction

The experimental dataset assembled for this study encompasses 3,000 URL samples, partitioned equally between 1,500 confirmed phishing instances and 1,500 verified legitimate entries. This balanced composition was deliberately chosen to prevent classifier bias toward the majority class — a common pitfall when class proportions diverge significantly. Malicious samples were drawn from PhishTank, a community-driven platform that aggregates verified phishing reports submitted by security researchers worldwide and refreshes its database on an hourly cadence, ensuring that collected samples reflect contemporary attack patterns rather than historical artefacts. The legitimate counterpart was sourced from a dataset published by the University of New Brunswick, which catalogues 35,300 benign URLs spanning a broad cross-section of domains and geographic origins.

Preparing the merged dataset for model consumption required three sequential operations. The first was data consolidation and sanitisation: the two source files were combined into a single unified table, after which all records containing null or undefined field values were identified and excised, as incomplete observations can introduce systematic bias during gradient-based optimisation or tree-splitting procedures. The second stage involved automated feature extraction, executed through Python's urlparse library for decomposing URL components and the whois module for querying domain registration metadata. The third and final stage was model training and evaluation, in which the extracted feature matrix was passed to each classifier under a consistent experimental protocol.

B. Feature Engineering

Fifteen discriminative features were derived from each URL sample and partitioned into two semantic categories: address bar characteristics, which concern the textual and structural properties of the URL string itself, and domain-level characteristics, which concern the administrative and reputational attributes of the hosting domain. Every feature is represented as a binary indicator variable, with the value 1 denoting a phishing-associated pattern and 0 indicating a benign characteristic.

1) Address Bar Characteristics

- Domain Token: The registered domain extracted from the full URL string. While informative for auditing purposes, this field carries limited generalisation value in isolation and is typically withheld from the training feature matrix to avoid overfitting on specific domain names.
- Embedded IP Address: Well-formed URLs reference destinations by domain name rather than numeric IP. When a raw IPv4 or IPv6 address appears in the URL body, it suggests an attempt to circumvent domain-reputation checks, warranting a positive phishing flag (1).
- At-Sign Inclusion: Browsers interpret the "@" delimiter as separating authentication credentials from the actual destination host; any content to its left is silently discarded. Phishing actors embed this character to mask the true destination behind a superficially plausible-looking prefix, triggering a value of 1.
- URL Character Length and Path Depth: Excessively long URLs are a recognised obfuscation strategy, burying the genuine landing domain within a string of decoy tokens. URLs surpassing 54 characters receive a phishing designation (1). Path depth — the count of forward-slash-delimited segments — serves as a complementary structural complexity measure.
- Anomalous Double-Slash Positioning: Protocol-conformant URLs place the "//" separator at character offset 6 (HTTP) or 7 (HTTPS). Discovery of this sequence at any other position signals an embedded redirect or protocol confusion, and is flagged as phishing.
- Protocol String Within Hostname: Inserting the literal text "http" or "https" inside the hostname segment — rather than in the scheme prefix — is a visual deception tactic designed to fool users who scan only the middle portion of a URL. Such occurrences are tagged as phishing (1).
- Hyphen Characters in Domain Label: While technically permissible, hyphens within domain labels are uncommon in legitimate brand domains and are disproportionately associated with typosquatting and brandjacking campaigns. Their presence yields a phishing indicator of 1.
- Appearance in Security Threat Reports: Organisations such as PhishTank and StopBadware maintain rolling lists of domains confirmed as participants in active phishing infrastructure. Membership in these lists at query time produces a value of 1.

2) Domain-Level Characteristics

- Absence of DNS Registration Record: A domain that cannot be resolved via WHOIS — the authoritative registry for domain ownership and contact metadata — has no verifiable legitimate identity. Such absence is treated as a strong phishing signal and assigned a value of 1.
- Alexa Traffic Ranking: Sustained web traffic, as indexed by the Alexa platform, correlates strongly with domain legitimacy. Domains ranked outside the top 100,000 globally — indicating negligible organic traffic — are considered suspect and receive a value of 1, since freshly minted phishing domains have not yet accumulated visitor history.
- Domain Registration Age: Phishing infrastructure is characteristically ephemeral; operators register domains immediately before a campaign and abandon them once detection probability rises. Domains with a registration age below twelve months are flagged with a value of 1.
- Proximity to Registration Expiry: A domain scheduled to expire within six months of the evaluation date reflects the short time horizons typical of phishing operators, who have no interest in maintaining long-term online presence. This condition is assigned a phishing value of 1.
- Excessive Sub-domain Nesting: A URL containing more than three period characters in its hostname component is syntactically consistent with deep sub-domain hierarchies used to embed a trusted domain name (e.g., paypal) as a non-authoritative prefix. This pattern is assigned a value of 1.

C. Classification Algorithms

1) Decision Tree

Decision Trees partition the input feature space by selecting, at each internal node, the attribute that most effectively reduces class impurity among the samples reaching that node. The recursive binary splitting continues until leaf nodes correspond to sufficiently pure class populations or a regularisation constraint is met. Two complementary impurity measures guide this process: Gini impurity, which quantifies the probability of misclassifying a randomly drawn sample according to the node's class distribution, and Shannon entropy, which measures the information content of the class distribution:

$$L\_Gini = \sum f\_i\ (1 - f\_i) \qquad (Eq.\ 1)$$

$$L\_Entropy = -\sum f\_i \cdot log_2(f\_i) \qquad (Eq.\ 2)$$

Here, f_i represents the proportion of samples belonging to class i at the node under consideration, and the summation spans all distinct class labels c. The algorithm selects the split that maximally reduces whichever impurity criterion is chosen.

2) Random Forest

Random Forest extends the single Decision Tree by training an ensemble of T trees, each on a bootstrapped subsample of the training data and each constrained to consider only a random subset of features at each split. This dual randomisation — over samples and features — induces diversity among the constituent trees, substantially reducing the variance that plagues individual trees without introducing a corresponding increase in bias. For classification, the ensemble output is determined by majority vote across all T trees. Feature importance scores are computed as the mean normalised impurity reduction attributable to each feature, aggregated across all nodes and all trees in which that feature appears:

$$RF\_importance(i) = (1/T) \sum_j norm\_fii_j \qquad (Eq.\ 3)$$

$$fii = \sum\_\{j:\ splits\ on\ i\}\ s\_j \cdot \Delta impurity\_j \qquad (Eq.\ 4)$$

where s_j denotes the fraction of all training samples that reach node j, and Δimpurity_j is the reduction in impurity achieved by the split at node j.

3) Logistic Regression

Logistic Regression models the posterior probability of class membership as a sigmoid transformation of a linear combination of input features, making it well-suited to binary classification problems where decision boundaries are approximately linear. The model is parameterised by a coefficient vector α of dimension M, corresponding to the M input predictors in vector a:

$$log\ [\ P(y=1 \mid a) / P(y=0 \mid a)\ ] = \alpha^T a \qquad (Eq.\ 5)$$

Parameter estimation proceeds by maximising the log-likelihood of the observed class labels under this model, typically via gradient-based optimisation. Despite its computational efficiency and the interpretability of its learned coefficients, logistic regression is fundamentally constrained by its linear decision surface; feature spaces characterised by intricate non-linear interactions between predictors are poorly served by this formulation.

4) Support Vector Machine

Support Vector Machine identifies the maximum-margin separating hyperplane between two classes in a high-dimensional feature space. Given a labelled training corpus:

$$\{\ (x_1, y_1), (x_2, y_2), \ldots, (x\_m, y\_m)\ \};\quad x\_i \in \mathbb{R}^d,\ y\_i \in \{-1, +1\} \qquad (Eq.\ 6)$$

the classifier seeks the hyperplane parameterised by normal vector a and offset b that maximises the geometric margin between the two classes while correctly labelling all training points. The classification decision for a new instance x is:

$$f(x) = sgn(a^T x + b),\quad a \in \mathbb{R}^d,\ b \in \mathbb{R} \qquad (Eq.\ 7)$$

The kernel trick allows SVM to implicitly operate in very high-dimensional or infinite-dimensional feature spaces by computing inner products through a kernel function rather than explicit coordinate transformations. This property makes SVM particularly powerful for URL-based phishing detection, where the discriminative signal is distributed non-linearly across many lexical and structural features.

IV. EXPERIMENTAL RESULTS AND DISCUSSION

A. Experimental Setup

To enable a fair and reproducible comparison across classifiers, all models were trained and evaluated on a common feature matrix derived from the UCI Machine Learning Repository phishing benchmark.

Thirty features spanning URL lexical properties, domain-level metadata, and HTML structural signals were extracted using the pipeline described in Section III. The dataset was divided into training and held-out test partitions via stratified sampling, a procedure that preserves the original class ratio within each split and thereby prevents the test set from being unrepresentative of the true data distribution. No hyperparameter tuning specific to any single classifier was performed, ensuring that observed performance differences reflect inherent algorithmic characteristics rather than optimisation artifacts.

B. Results and Comparative Analysis

Each of the three evaluated classifiers — Logistic Regression, SVM, and Random Forest — was assessed using classification accuracy as the primary metric, defined as the proportion of test samples assigned to the correct class. The results reveal a clear and consistent performance ordering:

1) Logistic Regression attained an accuracy of 91.8%. As anticipated for a linear classifier applied to a non-linearly separable problem, the model exhibited systematic difficulty in resolving ambiguous boundary cases where phishing and benign URLs share several overlapping surface features. Nevertheless, its computational frugality and interpretability make it a valuable baseline.

2) Support Vector Machine improved upon the logistic baseline with an accuracy of 94.2%. By projecting the feature space into a higher-dimensional representation via kernel functions, SVM successfully constructed a more expressive decision boundary. The cost of this improvement was a notably longer training duration, a trade-off that may be consequential in latency-sensitive deployment contexts.

3) Random Forest delivered the highest accuracy at 98.5%, outperforming both alternatives by a substantial margin. The ensemble's inherent resistance to overfitting — stemming from the decorrelated diversity of its constituent trees — appears particularly well-matched to the phishing detection feature space, where individual features carry partial and sometimes contradictory discriminative information that ensemble aggregation reconciles effectively.

These results align closely with broader patterns observed in the phishing detection literature: ensemble methods that pool evidence from multiple weak learners consistently achieve the best trade-off between bias and variance. The performance hierarchy (Random Forest > SVM > Logistic Regression) mirrors the hierarchy of model expressiveness, suggesting that the underlying classification problem possesses genuine non-linear structure that simpler models cannot fully exploit.

V. CONCLUSION

This study set out to evaluate whether supervised machine learning could provide a reliable, scalable solution to a cybersecurity problem that has persistently eluded rule-based remedies. The experimental evidence is unambiguous: machine learning classifiers, and ensemble methods in particular, are highly capable of distinguishing phishing URLs from legitimate ones across a rich, multi-dimensional feature space. Among the three algorithms benchmarked, Random Forest emerged as the clear frontrunner, achieving 98.5% classification accuracy — a figure that compares favourably with published results across the phishing detection literature.

Beyond raw accuracy, the study contributes a structured analysis of the feature categories most informative for detection: URL structural anomalies, domain registration metadata, and third-party reputation signals each provide complementary evidence that collectively enables robust classification. The findings affirm that no single feature or feature category is sufficient in isolation; it is the integration of diverse signals through an expressive model that delivers reliable performance across the full spectrum of phishing strategies currently employed by adversaries.

VI. FUTURE WORK

The results presented here, while encouraging, were obtained under idealised laboratory conditions using a static dataset. Translating laboratory performance into operational impact requires addressing several challenges that lie beyond the scope of the present study.

A. Browser-Integrated Real-Time Detection

The most immediate planned extension is the development of a Google Chrome browser extension that operationalises the trained Random Forest model as an always-on background service. Rather than requiring users to manually submit URLs for assessment, the extension will intercept each navigation event, extract the requisite feature set from the destination URL and its associated domain metadata, and invoke the classifier before the page content is rendered. Detection at this pre-render stage is significant: it prevents credential harvesting forms from ever appearing in the user's browser, rather than warning after the fact.

B. User-Facing Alert Mechanism

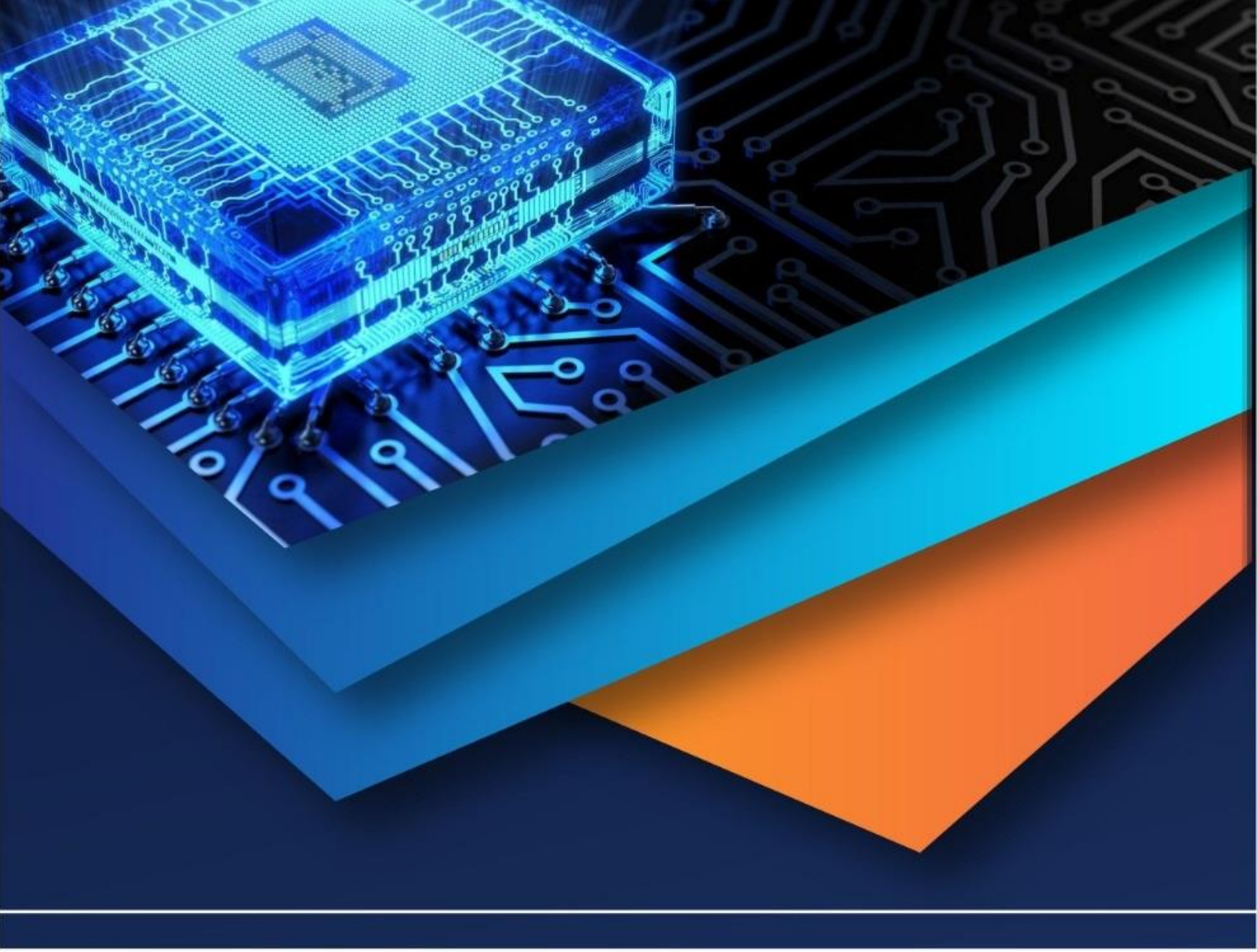
When the deployed model flags a URL as phishing, the extension will surface an interstitial warning screen that temporarily prevents further page interaction. The alert will clearly communicate the nature of the threat and offer the user actionable options — such as abandoning the navigation or reporting a suspected false positive — rather than passively logging the detection in a background process invisible to the user. Clear, timely feedback is essential to building the user trust that determines whether such a tool is actually adopted in practice.

C. Adaptive Model Maintenance

Static models trained on historical data gradually lose effectiveness as attackers observe detection patterns and adapt their URL construction strategies accordingly. To remain effective over time, the deployed extension must incorporate a mechanism for periodic model refreshment. The proposed architecture will support a pull-based update protocol in which the extension periodically fetches newly trained model weights from a remote server, trained on a continuously expanding corpus that incorporates recently confirmed phishing samples. This design separates the model lifecycle from the extension release cycle, enabling rapid response to emerging attack patterns without requiring users to install software updates.

REFERENCES

[1] R. Kiruthiga and D. Akila, "Phishing Websites Detection using Machine Learning," International Journal of Recent Technology and Engineering (IJRTE), vol. 8, no. 2, pp. 111–114, 2019.
[2] A. D. Kulkarni and L. L. Brown III, "Phishing Websites Detection using Machine Learning," ScholarWorks @ UT Tyler. [Online]. Available: https://scholarworks.uttyler.edu
[3] V. Shahrivari, M. M. Darabi, and M. Izadi, "Phishing Detection Using Machine Learning Techniques," arXiv preprint arXiv:2009.11116, 2020.
[4] Federal Bureau of Investigation, "2018 Internet Crime Report," Internet Crime Complaint Center (IC3), Washington, D.C., 2019.
[5] Microsoft Corporation, "Microsoft Computing Safer Index Report," Redmond, WA, 2014.
[6] D. M. Upadhyaya and M. A. Joshi, "Phishing – A new face of cyber crime," VSRD International Journal of CS & IT, vol. 2, no. 12, pp. 945–951, 2012.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY