Abstract – This document will provide a analysis of patent US11611582B2, which describes a computer-implemented method for detecting phishing threats. The analysis will cover various aspects of the patent, including its technical details, potential applications, and implications for cybersecurity professionals and other industry sectors.

Furthermore, it has a relevance to the evolving landscape of DevSecOps underscores its potential to contribute to more secure and efficient software development lifecycles as it offers a methodical approach to phishing detection that can be adopted by various tools and services to safeguard users and organizations from malicious online activities. Cybersecurity professionals should consider integrating such methods into their defensive strategies to stay ahead of emerging threats.

I. Introduction

The patent US20220232015A1 describes a method for dynamically detecting phishing threats using a pre-defined statistical model. This method is a machine learning based technique to dynamically analyze network requests in real-time and flag potential phishing attempts, in order to proactively protect users and systems from phishing attacks. The statistical model and feature set allow adapting to new phishing patterns.

II. Main idea

The main idea of the patent is to provide a scalable and automated approach to detect phishing attempts in real-time using machine learning, with the goal of proactively protecting users from falling victim to increasingly sophisticated phishing attacks. The dynamic analysis of web request attributes allows identifying new phishing sites that static lists may miss.

- The patent describes a computer-implemented method for dynamically detecting phishing threats using a pre-defined statistical model. The goal is to determine in real-time if a requested network resource is a potential phishing threat.
- When a request to access a network resource is received, a set of features associated with the request are extracted. These features may include the fully qualified domain name (FQDN), age of the domain, domain registrar, IP address, geographic location, etc.
- The extracted features are fed into a pre-trained statistical model which outputs a probability score indicating the likelihood that the requested resource is a phishing threat. If the score exceeds a pre-defined threshold, an alert is generated.
- The statistical model is trained using machine learning techniques on datasets containing known phishing and non-phishing examples. It can be periodically updated with new training data to adapt to evolving phishing patterns.

III. Industries

The specific implementation details and integration points will vary based on each industry's unique requirements and existing technology stack. However, the core capabilities of dynamic phishing detection using machine learning can be tailored to deliver significant security benefits in a wide range of sectors facing the growing threat of phishing attacks.

A. Telecommunications:

Telecom companies can integrate the phishing detection system into their network infrastructure to protect customers from phishing attacks delivered via SMS, MMS, or other messaging services.

The real-time detection capabilities can help block phishing links before they reach end-users, reducing the risk of account compromise and identity theft.

Telecom providers can offer phishing protection as a value-added service to differentiate themselves in the market and build customer trust.

B. Information Technology:

IT companies can deploy phishing detection solution as part of their cybersecurity offerings to clients, helping protect against phishing attacks targeting employees and customers.

Managed Security Service Providers (MSSPs) can integrate the technology into their threat monitoring and incident response services to detect and block phishing attempts in real-time.

Software-as-a-Service (SaaS) providers can embed the phishing detection into their platforms to scan for suspicious URLs and attachments, enhancing the security of their applications.

C. Finance:

Financial institutions can use the phishing detection system to protect their customers from targeted phishing attacks aimed at stealing login credentials, credit card numbers, and other sensitive financial data.

The solution can be integrated into online banking platforms, mobile apps, and email systems to scan for and flag potential phishing attempts in real-time.

By proactively detecting and blocking phishing threats, financial firms can reduce fraud losses, maintain customer trust, and comply with regulatory requirements for data protection.

D. Healthcare:

Healthcare organizations can leverage the phishing detection technology to safeguard sensitive patient data and prevent phishing attacks that could compromise the confidentiality, integrity, and availability of healthcare systems.

The solution can be deployed to monitor email communications, patient portals, and other digital channels for signs of phishing attempts targeting healthcare staff or patients.

By detecting and blocking phishing threats, healthcare providers can mitigate the risk of data breaches, protect patient privacy, and ensure the continuity of critical healthcare services.

E. E-commerce:

Online retailers can integrate the phishing detection capabilities into their e-commerce platforms to protect customers from phishing attacks that could lead to account takeover, fraudulent transactions, and identity theft.

The real-time detection can help identify and block phishing attempts delivered via fake order confirmation emails, account verification requests, or customer support inquiries.

By proactively addressing phishing threats, e-commerce companies can maintain customer trust, reduce chargebacks and fraud losses, and safeguard their brand reputation.

IV. The Proposed Solution

The key aspects are extracting relevant features from web requests, using a trained statistical model to score the requests, updating the model over time, and generating alerts when the score exceeds a threshold. This allows dynamic and adaptive detection of phishing threats. The key components of the method are:

Feature Extraction:

- When a request to access a network resource is received, a set of features associated with the request are extracted.
- These features may include the fully qualified domain name (FQDN), age of the domain, domain registrar, IP address, geographic location, etc.
- Feature extraction allows representing the key attributes of the web request that can indicate if it is a potential phishing attempt.

Statistical Model:

- The extracted features are fed into a pre-trained statistical model which outputs a probability score.
- The model is trained using machine learning techniques on datasets containing known phishing and non-phishing examples.
- Various ML models like logistic regression, decision trees, neural networks etc. can be used.
- The model learns the patterns and combinations of feature values that are indicative of phishing.

Model Training and Updating:

- The statistical model is initially trained on a labeled dataset before deployment.
- It can be periodically retrained with new training data to adapt to evolving phishing patterns.
- Updating the model allows it to recognize new phishing techniques and maintain accuracy over time.

Thresholding and Alert Generation:

- The output of the model is a probability score indicating the likelihood of the web request being a phishing attempt.
- If the score exceeds a pre-defined threshold, an alert is generated.
- The threshold can be adjusted to tune the sensitivity of the system based on desired false positive vs false negative rates.
- Protective actions can be taken like blocking the web request when an alert is triggered.

V. Process Flow

This process flow covers the end-to-end lifecycle of the proposed phishing detection system, from initial data collection and model development to real-time deployment, alert generation, and continuous improvement through model updates. The key stages are feature extraction, model training and evaluation, real-time scoring of live network requests, alert generation and response, and periodic model retraining to adapt to evolving phishing tactics.

Data Collection and Preprocessing:

- Collect a dataset of known phishing and legitimate network resource requests.
- Preprocess the raw request data to extract relevant features like URL, domain age, registrar, IP address, geographic location, etc.
- Label each request example as phishing or benign.

Feature Extraction:

- Define a set of discriminative features that can distinguish phishing attempts from legitimate requests based on domain knowledge and prior research.
- Implement feature extraction logic to parse the relevant attributes from the preprocessed request data.
- Transform the extracted feature values into a suitable format (e.g., numerical vectors) for input to the machine learning model.

Model Training:

- Select a machine learning algorithm for the phishing classification task (e.g., Random Forest, SVM, Neural Networks).
- Split the labeled dataset into training and testing subsets.
- Train the chosen model on the training set, learning the patterns that map the input features to the phishing/benign labels.
- Tune the model's hyperparameters using techniques like cross-validation to optimize performance.

Model Evaluation:

- Evaluate the trained model's performance on the held-out testing set.
- Calculate evaluation metrics such as accuracy, precision, recall, F1-score, etc.
- Analyze the model's performance to assess its effectiveness in detecting phishing attempts and identify areas for improvement.

Model Deployment:

- Integrate the trained phishing detection model into a live network monitoring system.
- Extract the same set of features from incoming network requests in real-time.
- Apply the model to each request's features to obtain a phishing probability score.
- Compare the score to a predefined threshold to classify the request as phishing or benign.

Alert Generation and Response:

- If a request's phishing score exceeds the threshold, generate an alert with relevant details like URL, source IP, risk score, etc.
- Deliver the alerts to security teams via appropriate channels like email, SMS, SIEM integration, etc.
- Trigger automated response actions based on alert severity, such as blocking the request or quarantining associated network traffic.
- Conduct manual investigation and remediation of high-priority alerts by security analysts.

Model Updating:

- Continuously collect new examples of phishing and benign requests in production.
- Periodically retrain the phishing detection model on an updated dataset to learn new attack patterns.
- Evaluate the retrained model's performance and deploy it to replace the existing model if it offers improved accuracy.
- Monitor the model's predictions over time to detect concept drift or performance degradation that may require further updates.

VI. Feature Extraction

Feature extraction involves identifying and selecting relevant characteristics or attributes from the raw data of the network resource request. The extracted features, such as FQDN, domain age, registrar, IP address and location, serve as inputs to the statistical model to dynamically assess phishing risk.

The goal is to transform the request data into a set of informative features that can be fed into the statistical model to determine if the request is potentially malicious.

- **Fully Qualified Domain Name (FQDN)**: The complete domain name of the requested resource, which includes the hostname, subdomain (if present), second-level domain, and top-level domain (TLD). For example, "mail.example.com" is an FQDN where "mail" is the hostname, "example" is the second-level domain and ".com" is the TLD.
- **Age of the Domain**: This refers to how long ago the domain name was registered. Newly registered domains are more likely to be associated with phishing attempts. The domain age can be determined by checking the domain's initial registration date.
- **Domain Registrar**: The entity through which the domain name was registered. Certain registrars may be more commonly used by phishing sites.
- **IP Address**: The numerical label assigned to the server hosting the requested resource.
- **Geographic Location**: The physical location of the server based on its IP address. Requests originating from unexpected geographic regions could indicate higher phishing risk.

Extracting these specific sub-features allows representing the key elements of the request in a structured format that can be analyzed by the statistical model. The feature values are likely transformed and normalized to make them suitable for input to the machine learning algorithm.

It is suggested that additional sub-features could also be extracted depending on the specific implementation. The feature extraction process essentially converts the raw request data into a vector of relevant attributes that succinctly capture the information needed to assess the phishing risk.

By carefully engineering and selecting the features, the accuracy and efficiency of the downstream phishing detection model can be optimized. The extracted features aim to capture patterns and signals that distinguish legitimate requests from phishing attempts based on the domain, server, and request characteristics.

VII. Statistical Model

The statistical model takes the extracted features of a network resource request as input and outputs a probability score

indicating the likelihood that the requested resource is a phishing threat.

Model Type: it suggests using machine learning techniques to train the statistical model, specifically mentioning the Random Forest algorithm as one possible implementation. Random Forest is an ensemble learning method that constructs multiple decision trees and outputs the class that is the mode of the classes output by the individual trees. It is known for its ability to generalize well to new data.

Model Inputs: The input to the model is the set of features extracted from the network resource request, such as the FQDN, domain age, registrar, IP address, geographic location, etc. These features are transformed and normalized into a suitable format (e.g. a feature vector) before being fed into the model.

Model Output: The output of the model is a probability score between 0 and 1, representing the estimated likelihood that the requested resource is a phishing attempt. If the score exceeds a predefined threshold (e.g. 0.8), the resource is classified as a potential phishing threat.2

Model Training: The statistical model is trained on a labeled dataset containing examples of known phishing and non-phishing (benign) network resources. During training, the model learns to recognize patterns and combinations of feature values that are indicative of phishing. The Random Forest algorithm adjusts the model parameters to minimize misclassification errors.

Model Evaluation: The performance of the trained model is evaluated using metrics like accuracy, precision, recall, F1-score, etc. on a separate test set. This helps assess how well the model generalizes to unseen data and guides model selection and hyperparameter tuning.

Model Updating: To adapt to evolving phishing tactics, the statistical model can be periodically retrained with new labeled data. This allows the model to learn new patterns and maintain its accuracy over time as the characteristics of phishing attempts change.

The statistical model is a machine learning classifier at the core of the dynamic phishing detection system. It is trained to predict the probability of a network resource being a phishing threat based on its extracted features. The model's architecture, training procedure, and updating strategy are designed to enable accurate, adaptive, and real-time identification of phishing attempts.

The use of a data-driven statistical approach allows the system to learn complex patterns from historical phishing data and generalize that knowledge to detect new, previously unseen phishing attempts. This provides a more dynamic and robust defense compared to static rule-based methods.

VIII. Model Training and Updating

Model training and updating refer to the processes of initially building the statistical model on a training dataset and subsequently refining it over time as new data becomes available. This is a crucial part of the machine learning pipeline that enables the phishing detection system to adapt and maintain accuracy in the face of evolving threats.

Initial Model Training:

- Before deployment, the statistical model (e.g., Random Forest classifier) is trained on a labeled dataset containing examples of known phishing and benign network resource requests.
- Each training example consists of the extracted features (FQDN, domain age, registrar, IP, location, etc.) and the corresponding label (phishing or benign).
- During training, the model learns to recognize patterns and combinations of feature values that distinguish phishing attempts from legitimate requests.
- The model's parameters are optimized to minimize prediction errors on the training data.

Periodic Model Updating:

- It emphasizes the importance of periodically retraining the model with new labeled data to adapt to evolving phishing tactics.1
- As new types of phishing attacks emerge, the characteristics of phishing requests may change over time.
- Updating the model allows it to learn these new patterns while retaining knowledge of previously seen phishing indicators.
- The frequency of model updates can be adjusted based on the volume and velocity of new phishing data collected.

Continuous Learning:

- Some machine learning architectures, like online learning or incremental learning, are specifically designed to support continuous updating of the model as new data arrives.
- Instead of retraining on the entire cumulative dataset, these methods can incrementally adjust the model parameters based on mini-batches of new examples.
- Continuous learning helps alleviate the computational burden of repeated retraining and allows faster adaptation to new threats.

Data Management:

- Effective model updating requires careful management of the training data over time.
- The labeled dataset needs to be expanded with new phishing and benign examples while maintaining a balance between the classes.
- Techniques like active learning can be used to strategically select the most informative examples for labeling, optimizing the use of human annotation efforts.

Evaluation and Monitoring:

- After each update, the retrained model should be evaluated on a separate test set to assess its performance and ensure it hasn't degraded.
- Continuous monitoring of the model's predictions in production is also important to detect concept drift or errors that may necessitate further updates.

The model training and updating are essential for the long-term effectiveness of the phishing detection system. The initial training process builds the model's baseline knowledge, while periodic updates allow it to adapt to new phishing patterns over time. Techniques like continuous learning, active data selection, and performance monitoring help optimize the update process and maintain the model's accuracy in the face of evolving threats.

IX. Thresholding and Alert Generation

Thresholding and alert generation refer to the process of deciding whether a given network resource request should be classified as a phishing attempt based on the probability score output by the statistical model, and raising an appropriate alert if the decision is positive. This is a critical step that translates the model's predictions into actionable security decisions and notifications.

Probability Score Threshold:

- The statistical model outputs a probability score between 0 and 1 for each network resource request, indicating the estimated likelihood of it being a phishing attempt.
- A predefined threshold value (e.g., 0.8) is used to make the final classification decision.
- If the score exceeds the threshold, the request is classified as a potential phishing threat. Otherwise, it is considered benign.

Threshold Selection:

- The choice of threshold value involves a trade-off between false positives (legitimate requests misclassified as phishing) and false negatives (phishing attempts misclassified as benign).
- A higher threshold reduces false positives but may miss some real phishing attempts. A lower threshold catches more phishing but also flags more benign requests.
- The optimal threshold can be determined based on the specific security requirements and the relative costs of false positives and false negatives in the deployment context.

Alert Generation:

- When a request's score exceeds the phishing threshold, an alert is generated to indicate a potential phishing threat.
- The alert may include relevant details about the request, such as the requested URL, source IP address, associated probability score, etc.
- Alerts can be delivered through various channels like console logs, email notifications, SMS messages, security incident and event management (SIEM) systems, etc.

Alert Validation and Filtering:

- To reduce false positives, generated alerts may go through additional validation steps before being escalated.
- This could involve comparing the alert details against allowlists of known benign resources, checking for alert flooding from the same source, or applying other heuristic filters.
- Manual review of a subset of alerts by security analysts can help tune the thresholds and validation rules over time.

Alert Response Actions:

- Depending on the severity and confidence of the phishing classification, different response actions can be triggered by the alerts.
- Lower severity alerts may simply be logged for later analysis, while higher severity ones may trigger immediate blocking of the resource request and quarantining of associated network traffic.
- Automated responses can be complemented by manual investigation and remediation actions based on the alert details.

The thresholding and alert generation bridge the gap between the probabilistic predictions of the phishing detection model and the deterministic security decisions and actions needed to protect users and systems. By selecting appropriate threshold values, generating informative alerts, and triggering proportional response actions, this component operationalizes the intelligence gathered by the model to provide effective anti-phishing defense.

X. Benefits, Drawbacks and Significance of Proposed Solution

This patent illustrates an important evolution from reactive, signature-based phishing detection to a more dynamic, adaptive approach powered by statistical modeling. While not a silver bullet, it represents a meaningful step towards stronger, more intelligent anti-phishing defenses.

This patent presents an automated, data-driven approach to detect phishing attempts in real-time by learning generalized patterns instead of using static rules. The dynamic nature allows adapting to evolving phishing techniques. Generating probabilistic risk scores enables prioritizing the most suspicious cases.

By describing a flexible machine learning pipeline with feature extraction, model training/updating, and alert generation, the patent provides a framework for building more effective anti-phishing systems. The proposed method could significantly improve an organization's ability to proactively identify and block phishing threats before they victimize users. However, it

does require substantial data collection and engineering effort to implement and maintain.

The statistical model is trained on historical phishing and benign examples to learn patterns that distinguish the two classes. It can be periodically retrained on new data to adapt to evolving phishing tactics.

Key Benefits:

- Enables proactive, real-time detection of phishing attempts, including new attacks not seen before, by analyzing patterns in URL/domain attributes
- Provides a probability score allowing prioritization of the riskiest threats
- Adapts to changing phishing tactics over time through periodic retraining of the model
- Generates informative alerts with key request details for security teams to investigate
- Allows tuning detection sensitivity by adjusting the alert threshold

Drawbacks:

- Requires significant historical phishing and benign data for initial model training
- Needs ongoing labeled data collection to retrain and update the model over time
- May miss some novel phishing patterns not reflected in the training data
- Extracting an effective feature set requires careful engineering and domain expertise
- Could generate false positives that may need additional filtering/validation

A. Feature Extraction

Feature extraction is a crucial step that allows building effective ML models for phishing detection by representing request data in an informative format. However, it requires significant expertise and effort to develop and maintain a robust feature set. Combining manual feature engineering with automatic representation learning can help alleviate some of these drawbacks and create more powerful hybrid detection models.

1) Benefits:

- Enables representing the key characteristics of network resource requests in a structured format suitable for analysis by machine learning models. Extracting relevant features is crucial for building accurate phishing detection models.
- Allows capturing discriminative patterns that distinguish phishing attempts from legitimate requests. Carefully engineered features can provide strong signals for classification.
- Reduces the dimensionality of raw request data, making it more computationally efficient to process. Working with a compact set of informative features is faster than analyzing the full request content.
- Feature extraction by domain experts leverages their knowledge to create highly relevant features for the specific task of phishing detection. Manual feature engineering guided by expertise can yield very effective feature sets.
- Extracted features can be combined with automatically learned features from deep learning to create powerful hybrid models. This allows getting the best of both manual feature engineering and representation learning.

2) Drawbacks:

- Requires significant domain expertise and manual effort to identify and implement effective features. Developing a good feature set for phishing detection is time-consuming and relies heavily on expert knowledge.
- Engineered features may not capture all relevant patterns, especially novel ones in evolving phishing attacks. There's a risk of missing important signals that experts haven't thought of.
- Feature extraction code needs to be regularly updated to handle changes in web technologies and phishing techniques. Maintaining the feature pipeline can be an ongoing engineering overhead.
- Extracted features may be specific to certain types of phishing attacks, limiting the model's ability to generalize to new attack variants. Overly specialized features can lead to brittle models.
- Relying solely on manually engineered features may result in lower performance compared to end-to-end deep learning on raw data. For some tasks, learned representations can outperform hand-crafted features.

B. Statistical Model

Statistical models, especially hybrid approaches combining engineered features and deep learning, offer powerful capabilities for dynamic and adaptive phishing detection. However, they also introduce challenges around data quality, feature engineering, computational complexity, and robustness to adversarial attacks. Effective deployment requires carefully addressing these limitations through continuous data collection, model updates, and expert oversight.

1) Benefits:

- Enables dynamic and adaptive detection of phishing threats by learning patterns from historical data. The statistical model can recognize complex combinations of features indicative of phishing, beyond simple rules.
- Outputs a probability score that quantifies the risk of a request being a phishing attempt. This provides more nuanced information than a binary classification, allowing fine-grained risk assessment and prioritization.
- Can be updated over time by retraining on new data to adapt to evolving phishing tactics. The model's

predictive power can be maintained as attackers change their techniques.

- Suitable for real-time detection due to fast inference time once the model is trained. Allows integration into live monitoring and prevention systems.
- Hybrid models combining manual feature engineering and deep learning have shown improved accuracy over traditional ML models in phishing detection. Leverages the strengths of both human expertise and data-driven learning.

2) *Drawbacks:*

- Requires a large labeled dataset for initial training, which can be expensive and time-consuming to obtain. Phishing datasets must be continuously updated to include new attack patterns.
- Model performance depends heavily on the quality and representativeness of the training data. Biased or incomplete datasets can lead to skewed predictions and blind spots.
- Feature engineering still plays a crucial role in building effective ML models for phishing detection. Relevant features must be manually crafted, requiring significant domain expertise.
- Traditional ML models like Random Forest may plateau in performance and fail to detect novel phishing patterns not seen during training. Keeping models up-to-date is an ongoing challenge.
- Deep learning models can be computationally expensive to train and may require specialized hardware. Increased complexity also makes the models harder to interpret and debug.
- Risk of adversarial attacks where phishers deliberately craft messages to evade detection by the model. ML models can be brittle and vulnerable to manipulation.

C. Model Training and Updating

The model training and updating are essential for maintaining the effectiveness of the phishing detection system as new threats emerge. However, the process also introduces operational complexities around data collection, labeling, computational resources, and change management. Careful design of the retraining pipeline, data quality controls, and monitoring mechanisms is crucial to realizing the benefits while mitigating the drawbacks.

1) *Benefits:*

- Allows the phishing detection model to adapt to evolving threats by learning from new labeled examples over time. Periodic retraining helps the model recognize novel phishing patterns.
- Continuous learning techniques can incrementally update the model with new data, reducing the computational cost compared to full retraining. This enables more frequent and efficient model updates.
- Active learning strategies can optimize the selection of new examples for labeling, minimizing the manual annotation effort required. This helps manage the ongoing data curation process.
- Regular model evaluation on new test sets ensures that updates actually improve performance and don't introduce regressions. Monitoring model behavior in production catches potential issues early.
- Updating the model with a diverse set of new phishing and benign examples improves its robustness and generalization to different attack variants. A broad training set helps the model handle a wide range of threats.

2) *Drawbacks:*

- Requires a continuous stream of new labeled phishing and benign examples to retrain the model, which can be challenging and expensive to obtain at scale. Labeling new training examples requires manual effort by domain experts and can be time-consuming. Developing efficient annotation workflows and interfaces is crucial.
- If the distribution of new training data differs significantly from the original data, the updated model may experience performance degradation or instability. Careful data quality control and monitoring are needed.
- Frequent model updates can be computationally expensive, especially for large deep learning models. Incremental learning techniques help but may still require significant resources.
- Updating the model changes its behavior, which can be disruptive to downstream systems and workflows relying on its predictions. Versioning and change management processes are important.
- There's a risk of the model overfitting to recent training examples and losing performance on older phishing patterns. Balancing the mix of old and new data during retraining is tricky.

D. Thresholding and Alert Generation

The thresholding and alert generation play a crucial role in operationalizing the phishing detection model by converting its probabilistic outputs into concrete security actions. However, this process also introduces challenges around threshold tuning, false positive management, and alert fatigue. Careful design and ongoing refinement of the thresholding logic, in tandem with the model's performance, are key to striking an effective balance between risk reduction and operational efficiency.

1) *Benefits:*

- Allows translating the probabilistic output of the statistical model into actionable security decisions. By comparing the model's phishing probability score to a predefined threshold, the system can automatically determine whether to flag a request as a potential threat.
- Provides a tunable parameter (the threshold) to balance the trade-off between false positives and false negatives. Adjusting the threshold allows administrators to control

the sensitivity of the alerts based on their risk tolerance and operational constraints.

- Enables generating informative alerts with relevant details about the suspicious request, such as the URL, source IP, and associated risk score. This contextual information helps security teams quickly triage and investigate potential phishing incidents.
- Supports flexible alert delivery channels, such as console logs, email, SMS, or integration with security information and event management (SIEM) systems. This allows phishing alerts to be seamlessly incorporated into existing security monitoring workflows.
- Allows implementing additional validation logic and filters to further reduce false positives. For example, alerts can be suppressed for whitelisted domains or IP ranges, or if the model's confidence score is below a certain level.

2) Drawbacks:

- Selecting an appropriate threshold value requires careful tuning and may involve trial and error. Setting the threshold too low can result in a high volume of false positives, while setting it too high may miss actual phishing attempts.
- The optimal threshold may need to be periodically adjusted as the characteristics of phishing attacks evolve over time. Maintaining an effective threshold requires ongoing monitoring and analysis of the system's performance and the changing threat landscape.
- Thresholding reduces the rich information provided by the model's probability score to a binary decision (alert or no alert). This can result in a loss of nuance and granularity in assessing the risk of borderline cases.
- Alerts generated by the system may still require manual review and investigation by security analysts. While thresholding helps prioritize the most suspicious cases, it doesn't completely eliminate the need for human judgment and intervention.
- The effectiveness of the alerts ultimately depends on the accuracy of the underlying statistical model. If the model's predictions are biased or miscalibrated, even a well-tuned threshold may produce suboptimal results.