Identificação de Ataques de *Phishing* através de *Machine Learning*

Bianca Domingos Guarizi[1], **Dalbert Matos Mascarenhas**[1]

[1]Centro Federal de Educação Tecnológica Celso Suckow da Fonseca - CEFET/RJ
Petrópolis - RJ - Brasil

bianca.guarizi@aluno.cefet-rj.br, dalbert.mascarenhas@cefet-rj.br

Abstract. *This work develops a Random Forest based tool that analyzes user traffic in the browser, identifies phishing sites, and alerts the user in real time. With 97.81% accuracy, the classifier was integrated with an extension that collects and classifies URLs in real time and alerts the user to malicious sites.*

Resumo. *Este trabalho desenvolve uma ferramenta baseada em Random Forest para analisar o tráfego do usuário no navegador, identificando sites de phishing e emitindo alertas em tempo real. Com 97,81% de acurácia, o classificador foi integrado a uma extensão que coleta e classifica URLs em tempo real, alertando o usuário sobre sites maliciosos.*

1. Introdução

Golpes de *phishing* são comuns no meio digital, mas a pandemia do Coronavírus intensificou esses ataques devido ao aumento das compras online [Pranggono and Arabo 2021, Al-Qahtani and Cresci 2022, Hoheisel et al. 2023]. Segundo a FEBRABAN, as tentativas de phishing cresceram 80% durante a quarentena [FEBRABAN 2020].

Atualmente existem diversas soluções disponíveis aos usuários que visam a proteção contra páginas de *phishing*. Extensões para navegador como Netcraft [Netcraft Ltda] e *Web of Trust* [WOT Services LP], soluções de antivírus como McAfee [McAfee], Kaspersky [Kaspersky Lab] e Norton [Gen Digital Inc.], além de proteções contra *phishing* integradas aos navegadores como o Google Safe Browsing [Google]. Entretanto, estas soluções se utilizam de bases de dados e listas negras, o que indica que para ataques *zero-day*, estas ferramentas se tornam ineficazes.

A ferramenta proposta executa a análise dos sites acessados em tempo real, através da utilização de um modelo de aprendizado de máquina que se comunica com uma extensão de navegador que é responsável pela coleta das URLs acessadas pelo usuário. O classificador implementado foi baseado no uso de *Random Forest* através da biblioteca scikit-learn, o qual retornou 97,81% de acurácia, possuindo melhores resultados em relação à trabalhos que se utilizaram do mesmo classificador.

O restante deste artigo está organizado da seguinte forma: A Seção 2 discute os trabalhos relacionados, a Seção 3 descreve sobre as tecnologias implementadas e como a ferramenta foi desenvolvida, a Seção 4 discute os resultados obtidos, enquanto a Seção 5 conclui este trabalho e descreve sobre trabalhos futuros.

2. Trabalhos Relacionados

Rodrigues *et al.* implementa três classificadores para detecção de *phishing*: J48, *Random Tree* e a rede neural WiSARD [Rodrigues and Bastos 2018], alcançando alta assertividade nos dois primeiros, com taxas de 94,18% e 93,21%, respectivamente e uma taxa inferior na última, com 81,8%. Além disso, foi utilizada uma base de apenas 13.726 URLs e nenhuma ferramenta final foi elaborada.

No trabalho de do Rego *et al.* foram utilizados os classificadores SVM, KNN e Regressão Logística para detecção de phishing [do Rego Cunha 2011]. Os modelos obtiveram 95,60%, 93,97% e 95,22% de acurácia, respectivamente. O trabalho também se utiliza de uma base pequena (12.912 URLs) e não elabora uma ferramenta funcional.

No trabalho de Correia *et al.*, foram testados quatro classificadores: Árvore de Decisão, Regressão Logística, SVM e *Random Forest* [Correia and Pedrini 2020]. O *Random Forest* apresentou o melhor desempenho, com acurácia final de 95%, porém o trabalho não desenvolve uma solução funcional.

Souza *et al.* executou a comparação entre MLP, SVM, KNN, Árvore de Decisão e *Random Forest* [Souza and Mascarenhas 2023]. Utilizando todos os atributos coletados, o melhor desempenho foi obtido através do classificador SVM, com acurácia de 99,19%, entretanto removendo atributos que proporcionavam mais tempo de resposta, o melhor desempenho foi obtido através do algoritmo *Random Forest* com 94,24%. O trabalho mostra boa performance, mas utiliza 990 URLs e não desenvolve uma ferramenta funcional.

No presente trabalho, o objetivo é a elaboração de um algoritmo de detecção que colete características de forma automatizada das páginas e as utilize para detecção de *phishing*. Além disso, o diferencial deste trabalho em relação aos apresentados acima é a elaboração de uma ferramenta para detecção em tempo real que possa ser utilizada pelos usuários.

3. Solução Proposta

3.1. Tecnologias Implementadas

Para elaboração do classificador, foi utilizada a linguagem de programação Python, juntamente com a biblioteca scikit-learn [Kramer and Kramer 2016]. O modelo de classificador escolhido foi o *Random Forest*, uma vez que este foi o que obteve melhor desempenho em Correia *et al.* e, após ajustes de remoção de atributos, também em Souza *et al.*. Além disso, a linguagem de programação JavaScript também foi utilizada para criação de uma extensão para o navegador que coleta as URLs acessadas e as envia ao classificador.

A comunicação entre a extensão de navegador e o código em Python é feita por um servidor central que é acessado utilizando a biblioteca Flask do Python, que acessa a rota de comunicação e gerencia o envio e recebimento de dados em formato JSON.

3.2. Ferramenta Desenvolvida

O classificador foi modelado de forma a prevenir a ocorrência de *overfitting* com os seguintes parâmetros: profundidade máxima da árvore de 10 (max_depth=10), divisão mínima de 10 amostras (min_samples_split=10), folhas mínimas de 4 amostras (min_samples_leaf=4), 100 árvores de decisão (n_estimators=100), e uso da raiz quadrada

dos recursos em cada divisão (max_features='sqrt'). Além disso, foi definido um critério mínimo de divisão dos nós (min_impurity_decrease=0.001) e o uso de bootstrap (bootstrap=True) para criar árvores em subconjuntos diferentes dos dados.

O fluxograma demonstrado na Figura 1 ilustra de forma mais abrangente as ações executadas por cada um dos componentes que integram a ferramenta elaborada.

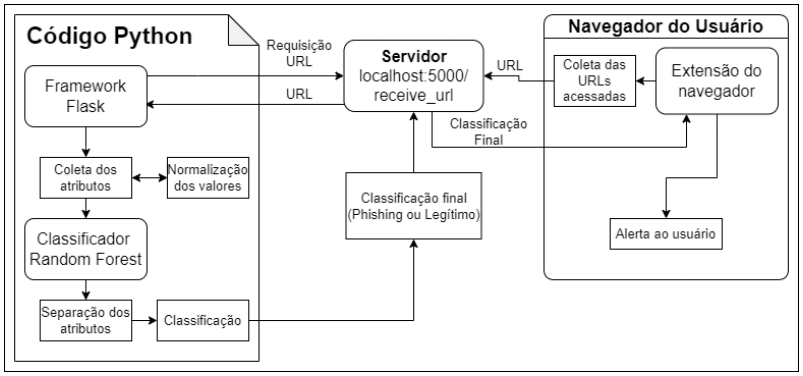

Figura 1. Fluxograma sobre as funcionalidades dos componentes da ferramenta

O código em Python treina o modelo de classificação e analisa cada URL acessada pelo usuário. Após coletar e normalizar os atributos, o modelo classifica o site como legítimo (0) ou *phishing* (1) e envia um alerta ao servidor. A extensão para navegador se encontra disponível para integração com navegadores Google Chrome e tem por objetivo analisar o tráfego, coletando as URLs acessadas pelo usuário. O servidor central que recebe as URLs coletadas pela extensão, as envia para classificação pelo código em Python e retorna o resultado à extensão, que exibe um alerta ao usuário.

3.3. Base de Dados

Para coleta das URLs de *phishing*, foram utilizadas 2 (duas) bases de dados públicas: PhishTank [Cisco Systems, Inc.] e OpenPhish [OpenPhish], que possuem uma ampla quantidade de sites de *phishing* registrados.

As URLs legítimas foram coletadas a partir de duas abordagens. Primeiro, foram acessados sites amplamente utilizados, como Google, LinkedIn e Amazon, para coletar as URLs. Em seguida, aplicou-se a técnica de enumeração de domínios com a ferramenta WayBack Urls [Hudson, Tom], utilizando a base de dados disponibilizada pelo site Moz [Moz, Inc.], para identificar domínios e subdomínios associados às URLs. Isto ampliou a base de URLs legítimas e ao mesmo tempo garantiu a legitimidade das URLs.

Juntando as abordagens descritas anteriormente, foi possível realizar a criação de uma base de dados de URLs legítimas consistente e ampla, diferente das bases disponíveis que geralmente são pequenas e não refletem a realidade da navegação executada pelo usuário no dia a dia. Ao fim, foi possível criar uma base de dados com um total de 50.261 URLs, com aproximadamente 55.5% de URLs de *phishing* e 44.5% de URLs legítimas.

3.4. Atributos Utilizados

Para treinamento do classificador elaborado, foram selecionados 13 atributos, demonstrados na Tabela 1, identificados durante os estudos de trabalhos relacionados e selecionados por sua associação com a distinção entre sites de *phishing* e legítimos.

Tabela 1. Tabela de atributos selecionados

Atributo	Descrição
A1 - Tamanho da URL	Sites maliciosos podem conter muitos caracteres para enganar o usuário e esconder informações suspeitas.
A2 - Utiliza HTTPS	Sites de *phishing* podem não utilizar protocolo HTTPS, enquanto sites legítimos geralmente o utilizam.
A3 - Possui formato IP	Sites de *phishing* podem não utilizar DNS, enquanto sites legítimos geralmente utilizam.
A4 - Quantidade de pontos (.)	Muitos pontos podem ser usados para adicionar subdomínios e fazer o domínio de *phishing* parecer legítimo.
A5 - Contém o caractere '@'	Esse caractere pode fazer com que o navegador ignore as informações anteriores à ele, permitindo ao atacante incluir um domínio real para enganar o usuário.
A6 - Total de dias ativo	Domínios com uma data de criação recente podem ter sido criados para *phishing*, já que ataques deste tipo geralmente duram pouco tempo.
A7 - *PageRank*	Páginas de *phishing* possuem pontuações consideravelmente baixas em relação a sites legítimos em pontuações de *PageRank*.
A8 - Possui formulário HTML	Páginas de *phishing* geralmente incluem formulários para capturar informações do usuário, mas outros atributos ajudam a diferenciá-las de páginas legítimas.
A9 - Validade do Certificado SSL	Sites de *phishing* geralmente usam certificados gratuitos, possuindo validade menor que certificados pagos.
A10 - Contém caracteres '//'	Quando usado, exceto em 'https://' ou 'http://', pode redirecionar o usuário para a URL após esse caractere.
A11 - Possui texto 'https' na URL	Isso pode enganar o usuário a pensar que o site possui certificado SSL.
A12 - Possui caractere '-' no domínio	Esse caractere pode enganar o usuário, separando palavras para simular uma página legítima.
A13 - Possui iframe no código HTML	Essa tag pode ser usada para incorporar uma página real em um site malicioso, enganando o usuário.

4. Resultados

4.1. Descrição dos resultados

Para realizar a avaliação final sobre a ferramenta desenvolvida, foram utilizadas 4 métricas: acurácia, precisão, *recall* (sensibilidade) e F1 score. Os resultados obtidos para as métricas de acurácia, precisão e *recall* são mostradas na Matriz de Confusão demonstrada na Figura 2, tendo sido estes resultados obtidos através da separação dos dados em: 85% para treinamento e 15% para teste. Além dos testes realizados utilizando a base de dados elaborada, também foi coletada a base de dados criada por Souza *et al.*, com o intuito de verificar como o algoritmo se comporta com uma base de dados distinta.

É possível verificar que a precisão referente às classificações de páginas legítimas foi de 99,02% e para páginas de *phishing* foi de 96,90%, enquanto o *recall* sobre as classificações de páginas legítimas retornou 96,04% e para páginas de *phishing* 99,24%, por fim a acurácia obtida foi de 97,81%. Em relação à base de Souza *et al.*, o algoritmo foi capaz de retornar uma acurácia bastante próxima (98,66%), sendo melhor que o resultado obtido pelos autores (Este resultado pode ser reproduzido através dos códigos disponibilizados em [Guarizi and Mascarenhas 2024]). Além disso, a última métrica analisada foi o F1 Score, tendo como resultado 97,50%. A fim de verificar se o modelo pode estar apresentando *overfitting*, também foram analisadas estas métricas sobre o conjunto de treinamento.

Figura 2. Matriz de Confusão

Tabela 2. Tabela de comparação de métricas

	Precisão	***Recall***	**Acurácia**	**F1 Score**
Resultado - Teste	99.02%	96.04%	97.81%	97.50%
Resultado - Treinamento	98.95%	96.38%	97.93%	97.65%

As métricas de treinamento e teste semelhantes, demonstradas na Tabela 2, indicam que o modelo generaliza bem, com previsões satisfatórias no conjunto de teste, que era desconhecido durante o treinamento. Também foram testados o tempo de treinamento e o impacto na CPU e RAM. O treinamento levou 2,40 segundos, com uso médio de 0,2% da CPU e RAM. O tempo de resposta durante a navegação foi de 1,6 segundos, em média.

5. Conclusão

É possível perceber que, mesmo com menos atributos coletados do que em outros projetos com o mesmo intuito como Souza *et al.* e Correia *et al.*, o classificador elaborado alcançou resultados satisfatórios de modo que os trabalhos em questão obtiveram acurácias de 94,24% e 95%, respectivamente, utilizando *Random Forest* e o presente trabalho alcançou 97,81% de acurácia, porém o algoritmo retornou um desempenho menos significativo quando comparado ao SVM em Souza *et al.*, com 99,19% de acurácia. Além disso, o presente trabalho implementou uma base de dados mais robusta para treinamento do modelo em relação a trabalhos relacionados que se utilizaram de bases de dados menores.

Este trabalho desenvolveu uma ferramenta baseada no algoritmo *Random Forest* para identificar possíveis sites de *phishing*. A ferramenta analisa automaticamente as páginas acessadas e alerta o usuário sobre a segurança da navegação por meio de um pop-up, que requer confirmação para prosseguir. Em trabalhos futuros, uma implementação menos intrusiva pode ser considerada, com alertas que desaparecem após alguns segundos, sem interromper a navegação. Além disso, será possível adicionar mais atributos para melhorar a classificação e realizar testes com usuários reais, focando na redução do tempo de resposta e aprimorando a experiência do usuário.

Referências

Al-Qahtani, A. F. and Cresci, S. (2022). The covid-19 scamdemic: A survey of phishing attacks and their countermeasures during covid-19. *IET Information Security*, 16(5):324–345.

Cisco Systems, Inc. Phishtank. Disponível em `https://phishtank.org/`. Acessado em maio de 2024.

Correia, P. H. B. and Pedrini, H. (2020). Detecção de domínios maliciosos baseada em técnicas de aprendizado de máquina. Trabalho de conclusão de curso (Bacharelado em Ciência da Computação). Universidade Estadual de Campinas.

do Rego Cunha, F. F. (2011). Detecção de phishing em páginas web. Programa Institucional de Bolsas de Iniciação Científica. Universidade Federal do Amazonas.

FEBRABAN (2020). Conheça as tentativas de golpes financeiros mais comuns na pandemia e saiba como evitá-los. Disponível em `https://portal.febraban.org.br/noticia/3522/pt-br/`. Acessado em maio de 2024.

Gen Digital Inc. Norton. Disponível em `https://br.norton.com/`. Acessado em agosto de 2024.

Google. Google safe browsing. Disponível em `https://safebrowsing.google.com/`. Acessado em agosto de 2024.

Guarizi, B. D. and Mascarenhas, D. M. (2024). Findphishing. Disponível em `https://github.com/bguarizi/findphishing/`. Acessado em julho de 2024.

Hoheisel, R., van Capelleveen, G., Sarmah, D. K., and Junger, M. (2023). The development of phishing during the covid-19 pandemic: An analysis of over 1100 targeted domains. *Computers & Security*, 128:103158.

Hudson, Tom. Wayback urls. Disponível em `https://github.com/tomnomnom/waybackurls`. Acessado em junho de 2024.

Kaspersky Lab. Kaspersky. Disponível em `https://www.kaspersky.com.br/`. Acessado em agosto de 2024.

Kramer, O. and Kramer, O. (2016). Scikit-learn. *Machine learning for evolution strategies*, pages 45–53.

McAfee, L. Mcafee. Disponível em `https://mcafee.com/pt-br/index.html`. Acessado em agosto de 2024.

Moz, Inc. The moz top 500 websites. Disponível em `https://moz.com/top500`. Acessado em junho de 2024.

Netcraft Ltda. Netcraft. Disponível em `https://www.netcraft.com/`. Acessado em agosto de 2024.

OpenPhish. Openphish. Disponível em `https://openphish.com/`. Acessado em maio de 2024.

Pranggono, B. and Arabo, A. (2021). Covid-19 pandemic cybersecurity issues. *Internet Technology Letters*, 4(2):e247.

Rodrigues, L. F. D. F. and Bastos, I. A. M. D. M. (2018). Um sistema inteligente para prevenção de ataques phishing. Trabalho de conclusão de curso (Bacharelado em Engenharia de Redes de Comunicação). Universidade de Brasília.

Souza, J. A. and Mascarenhas, D. M. (2023). Detecção de ataques de phishing em tempo real utilizando algoritmos de aprendizado de máquina. In *Anais Estendidos do XXIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais*. SBC.

WOT Services LP. Wot (web of trust). Disponível em `https://www.mywot.com/pt`. Acessado em agosto de 2024.