UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA
CIÊNCIAS DA COMPUTAÇÃO

Antonio Silverio Montagner

Sistema de Detecção de Requisições WEB Maliciosas

Florianópolis
2023

Antonio Silverio Montagner

Sistema de Detecção de Requisições WEB Maliciosas

Trabalho de Conclusão de Curso submetido ao Curso de Graduação em Ciências da Computação do Centro Tecnológico da Universidade Federal de Santa Catarina como requisito para obtenção do título de Bacharel em Ciências da Computação.

Orientadora: Profa. Carla Markle Westphall, Dra.
Coorientador: Rômulo Augusto Oliveira Cruz Bittencourt de Almeida, Mr.

Florianópolis
2023

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Montagner, Antonio Silverio
Sistema de Detecção de Requisições WEB Maliciosas / Antonio Silverio Montagner ; orientadora, Carla Markle Westphall, coorientador, Rômulo Augusto Oliveira Cruz Bittencourt de Almeida, 2023.
92 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Centro Tecnológico, Graduação em Ciências da Computação, Florianópolis, 2023.

Inclui referências.

1. Ciências da Computação. 2. Detecção de Phishing. 3. Ataque Cibernético. 4. Antiphishing. 5. Detecção de Phishing Baseada em DNS. I. Westphall, Carla Markle. II. Almeida, Rômulo Augusto Oliveira Cruz Bittencourt de. III. Universidade Federal de Santa Catarina. Graduação em Ciências da Computação. IV. Título.

Antonio Silverio Montagner

Sistema de Detecção de Requisições WEB Maliciosas

Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do Título de Bacharel em Ciências da Computação e aprovado em sua forma final pelo curso de Graduação em Ciências da Computação.

Florianópolis, 01 de Dezembro de 2023.

Profa. Jerusa Marchi, Dra. &
Profa. Lucia Helena Martins Pacheco, Dra.
Coordenadoras do Curso

Banca Examinadora:

Profa. Carla Markle Westphall, Dra.
Orientadora
Universidade Federal de Santa Catarina

Rômulo Augusto Oliveira Cruz Bittencourt de Almeida, Mr.
Coorientador
Universidade Federal de Santa Catarina

Profa. Thaís Bardini Idalino, Dra.
Avaliador
Universidade Federal de Santa Catarina

Guilherme Eliseu Rhoden, Mr.
Avaliador
Rede Nacional de Ensino e Pesquisa

Este trabalho é dedicado à minha familia e a todos que fizeram
parte da minha jornada.

AGRADECIMENTOS

Agradeço, primeiramente, aos meus pais, irmãos e familiares dos quais fizeram parte de minha educação e deram suporte aos meus sonhos. Também agradeço a minha namorada Rafaela Tillmann por me apoiar durante esta caminhada. Agradeço a professora Carla Markle Westphall por me aceitar como orientando e por me fornecer suporte no desenvolvimento deste trabalho. Agradeço também ao Rômulo Almeida por aceitar me coorientar neste trabalho. Agradeço também a Profa. Thaís Bardini Idalino e ao Guilherme Eliseu Rhoden por aceitarem fazer parte de minha banca avaliadora. E, por fim, agradeço a todos os professores que fizeram parte de minha caminhada de graduação na UFSC e a todos os amigos que fiz nesse caminho.

"Na grande jornada da vida, a verdadeira sabedoria reside em contemplar com discernimento nossas ações, traçar com clareza o nosso percurso e, com paciência, dedicar a cada passo o seu devido tempo. Dessa maneira, forjamos nosso destino com um propósito firme, em que, a cada passo, encontramos não apenas uma conquista, mas também uma oportunidade de aprendizado." (Eu)

RESUMO

O *phishing*, uma técnica de engenharia social pela qual os atacantes exploram os pontos fracos das vítimas para obter informações confidenciais, é um problema que assola a sociedade há mais de duas décadas. Com a constante evolução das tecnologias de comunicação e o aumento da interatividade entre as pessoas, as táticas de ataque cibernético, incluindo o *phishing*, também se aprimoraram. Essa sofisticação levou a um aumento na eficácia desse tipo de ataque, tornando-o uma das ameaças mais proeminentes enfrentadas pelos usuários da Internet. A persistência desse desafio resultou em pesquisas dedicadas à criação de soluções eficazes para detectar e mitigar o *phishing*. Esses esforços se tornaram cruciais à medida que o número de vítimas desse tipo de ataque continuou a crescer ao longo dos anos. Assim, a comunidade de pesquisadores trabalha incansavelmente para desenvolver métodos e ferramentas capazes de identificar características específicas dos ataques de *phishing*, com o objetivo de prevenir a efetividade dessas ameaças e proteger os usuários.

O propósito deste trabalho é aprofundar a compreensão do *phishing* e suas táticas de ataque, avaliar as soluções existentes para a detecção dessas técnicas e, por fim, desenvolver um método para identificar domínios associados ao *phishing*. Para alcançar esse objetivo, será aplicado um estudo aprofundado sobre o estado da arte em relação ao *phishing*, abrangendo uma revisão das tecnologias e abordagens relevantes, assim estabelecendo uma base sólida para a implementação de um método de detecção de domínios de *phishing*. A contribuição deste trabalho não se limita à teoria, mas se estende à prática. Propomos o desenvolvimento e demonstração de um sistema de detecção de *phishing* sob uma arquitetura ARM, projetado para facilitar a implantação por meio da tecnologia de virtualização em contêineres. Este sistema incorpora uma estratégia de detecção que utiliza logs de um servidor DNS como parte integral do processo de identificação de tentativas de *phishing*. Com isso, esta iniciativa visa contribuir com o desafio crescente da cibersegurança, oferecendo uma abordagem robusta e prática para a identificação e prevenção de ataques de *phishing*, assim promovendo um ambiente digital mais seguro e confiável.

Palavras-chave: Detecção de Phishing, Ataque Cibernético, Antiphishing, Detecção de Phishing Baseada em DNS.

ABSTRACT

"Phishing", a social engineering technique through which attackers exploit victims' vulnerabilities to obtain confidential information, has been a problem plaguing society for over two decades. With the continuous evolution of communication technologies and increased interactivity among individuals, cyber attack tactics, including phishing, have also advanced. This sophistication has led to an increase in the effectiveness of such attacks, making it one of the most prominent threats faced by internet users. The persistence of this challenge has spurred dedicated research into the creation of effective solutions for detecting and mitigating phishing. These efforts have become crucial as the number of victims of such attacks has continued to grow over the years. Thus, the research community works tirelessly to develop methods and tools capable of identifying specific characteristics of phishing attacks to prevent the effectiveness of these threats and protect these users.

The purpose of this work is to deepen the understanding of phishing and its attack techniques, evaluate existing solutions for detecting these techniques, and ultimately develop a method for identifying phishing-related domains. To achieve this goal, a thorough study of the state of the art in phishing will be conducted, encompassing a review of relevant technologies and approaches, thus establishing a solid foundation for implementing a phishing domain detection method. The contribution of this work goes beyond theory and extends to practice. We propose the development and demonstration of a phishing detection system under an ARM architecture, designed for easy deployment through Docker technology. This system incorporates an innovative detection strategy by making use of DNS server logs as an integral part of the phishing attempt identification process. This initiative aims to address the growing challenge of cybersecurity by offering a robust and practical approach to the identification and prevention of phishing attacks, thereby promoting a safer and more reliable digital environment.

Keywords: Phishing Detection, Cyberattack, Anti-phishing, DNS Based Phishing Detection.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

LGPD	Lei Geral de Proteção de Dados Pessoais
GDPR	General Data Protection Regulation
URL	Uniform Resource Locator
DNS	Domain Name Server
IP	Internet Protocol
RNP	Rede Nacional de Ensino e Pesquisa
POP – SC	Ponto de Presença da RNP em Santa Catarina
ISA	Instruction set architecture
RTT	Round Trip Time
CPU	Central Processing Unit
RAM	Random Access Memory

SUMÁRIO

1 INTRODUÇÃO

O termo *phishing* é uma variação do termo inglês *fishing*, ou seja, o ato de *phishing* se assemelha ao de pescar, onde o atacante utiliza uma isca para atrair a vítima (CHIEW; YONG; TAN, 2018). Esse tipo de ataque está presente no mundo cibernético há mais de duas décadas e vêm crescendo ao longo dos anos. Atualmente uma grande quantidade de ataques de *phishing* exploraram, entre outros cenários, o de pandemia da COVID-19 (ABROSHAN et al., 2021), onde os indivíduos ficaram mais dependentes de serviços onlines, como aplicativos de mensagens e de mídias sociais, aumentando o número de técnicas de abordagem que podem se usadas no ataque, assim possibilitando uma maior efetividade.

Tal tipo de ataque pode causar danos às vítimas por ser um ataque de engenharia social, que pode explorar características sociais e psicológicas do alvo, para retirar traços pessoais que podem ser cruciais na tomada de uma decisão por essa pessoa. Como exemplo, o atacante pode utilizar um e-mail, como vetor, enviado pela Internet, como meio, e usar uma abordagem específica para seu alvo baseado em detalhes sociais e psicológicos coletados, assim, podendo se passar por alguma instituição e oferecer um produto específico para mascarar seu ataque. Dessa forma, fica difícil para a vítima notar que tal e-mail não é legítimo e pode baixar algo malicioso ou fornecer algum dado que seja pedido pelo atacante (SYAFITRI et al., 2022).

Tendo em vista tais problemas, pesquisas vêm desenvolvendo algumas formas para prevenção e detecção deste tipo de ataque cibernético, tal como a forma de detecção de ataques desenvolvidos na dissertação de mestrado descrita em (ALMEIDA, 2022). Nessa dissertação, por meio de um método verificação de strings, são detectadas presenças de alguns elementos que buscam contato direto com o usuário, tal como links de redirecionamento e arquivos para download, que são elementos importantes para a implementação de uma abordagem em um ataque de *phishing*.

Este trabalho tem como proposta realizar uma investigação sistemática do estado da arte do *phishing* e de seus métodos de detecção. Além disso, busca desenvolver um exemplo de software voltado para a detecção de domínios associados ao *phishing* e implementá-lo em um ambiente de testes controlado. Essa implementação possibilitará a coleta de dados que será posteriormente analisada, contribuindo assim para um entendimento mais aprofundado do *phishing* e suas técnicas de detecção. Os resultados deste estudo têm o potencial de servir como base para futuros trabalhos na área de cibersegurança.

Sendo assim, o presente trabalho propõe o desenvolvimento e demonstração de um sistema de detecção de *phishing* projetado sob uma arquitetura ARM. A implementação simplificada é facilitada pelo uso da tecnologia de virtualização em contêineres, tornando a implantação mais eficiente e prática. Esse sistema utiliza o registro de logs de um servidor DNS para estabelecer uma estratégia de detecção em conjunto com um banco de dados de URLs maliciosas implementado por uma aplicação desenvolvida em Python.

1.1 MOTIVAÇÃO

Após mais de 25 anos na sociedade, tendo surgido em 1995 (JAMES, 2006), o número de registros de URLs exclusivas de *phishing* registrados no final de 2022 alcançou mais de 450 mil (APWG, 2022). Esses ataques exploraram, entre outros cenários, o da pandemia da COVID-19 (ABROSHAN et al., 2021), onde os indivíduos ficaram mais dependentes de serviços onlines, como aplicativos de mensagens e de mídias sociais. Com isso, a conformidade desses serviços com a LGPD (BRASIL, 2018) e GDPR (UNIÃO EUROPÉIA, 2016) desempenham um papel vital na proteção da privacidade dos dados, reduzindo o risco de violações de segurança e prevenindo a exposição de informações pessoais de seus usuários.

Nesse contexto, deve-se considerar o fator humano relacionado com o ataque de *phishing*, que é explorado por formas de engenharia social para aumentar a efetividade do ataque. Com isso, dependendo das ações do usuário, tal ponto pode custar o sucesso ou a falha do ataque (DESOLDA et al., 2021). Dessa forma, as pessoas se tornam alvos que são analisados em âmbito social e psicológico e, assim, a sensação de liberdade, privacidade e segurança dessas pessoas são comprometidas.

1.2 JUSTIFICATIVA

Dado o número de ataques *phishing*, que atingiu em 2022 o maior número de todos os tempos até então, sendo quatro vezes maior que o número alcançado em 2019 (APWG, 2022), e a data de seu surgimento, demonstra que os ataques de *phishing* se adaptam com o tempo, assim resistindo as evoluções da sociedade.

Com isso, explorando o fator humano por meio de técnicas de engenharia social, pessoas mal-intencionadas aproveitam de tais conhecimentos para desenvolver ataques, como o de *phishing*, em busca de benefício próprio e não se importando em afetar a vida dos outros.

Dessa forma, trabalhos como (ALMEIDA, 2022) e (ATIMORATHANNA et al., 2020) buscam pesquisar e implementar uma forma eficaz de detecção de *phishing* em prol de uma maior segurança cibernética. Seguindo o exemplo desses trabalhos, o presente trabalho também busca colaborar com a segurança cibernética, mais especificamente, com as formas de detecção de *phishing*.

1.3 OBJETIVOS

Nesta seção serão definidos os objetivos que este trabalho pretende alcançar, como os objetivos gerais, descrevendo de maneira ampla o escopo de trabalho, e os objetivos específicos, descrevendo mais especificamente o que pretende ser alcançado.

1.3.1 Objetivos Gerais

Levando em conta os problemas previamente apresentados, o objetivo geral é conduzir um estudo abrangente sobre o *phishing*, compreendendo desde seus conceitos fundamentais até os diversos tipos e os impactos que ele causa na sociedade. Além disso, serão exploradas as diferentes abordagens de detecção já tratadas na literatura, fornecendo uma visão geral das soluções existentes. O trabalho não se limita à teoria; visa também apresentar um exemplo de implementação prática, baseando-se na abordagem desenvolvida por (ALMEIDA, 2022), para detectar domínios de *phishing* em um ambiente controlado.

O presente trabalho propõe o desenvolvimento e demonstração de um sistema de detecção de *phishing* análisando registros de logs domínios gerados por um servidor DNS e verificando os domínios em uma base de dados de URLs maliciosas.

1.3.2 Objetivos Específicos

A seguir são listados os principais objetivos específicos deste trabalho:

1. Realizar uma revisão bibliográfica sobre o estado da arte de *phishing*;
2. Apresentar as vulnerabilidades e ameaças existentes na sociedade por causa de ataques de *phishing*;
3. Apresentar os métodos de segurança da informação aplicáveis para a detecção de domínios maliciosos;
4. Apresentar a proposta de desenvolvimento de um software para detecção de domínios relacionados a *phishing*;
5. Apresentar o ambiente e as tecnologias para a implementação dos testes;
6. Apresentar o processo de desenvolvimento e implementação do software desenvolvido;
7. Sintetizar os dados coletados e estabelecer uma conclusão sobre o conteúdo apresentado.

1.4 MÉTODO DE PESQUISA E TRABALHO

Para o desenvolvimento da solução, os seguintes métodos de pesquisa e trabalho serão utilizados:

1. O embasamento teórico acerca do *phishing*, suas modalidades de ataque, ameaças associadas e métodos de detecção será consolidado por meio de pesquisas em artigos científicos e publicações acadêmicas;

2. O método de pesquisa possui a revisão da literatura, coleta de dados, análise dos dados e apresentação da revisão bibliográfica feita;

3. A pesquisa destinada ao desenvolvimento do software proposto se alicerçará nas tecnologias estudadas durante a fase de embasamento teórico, pressupondo-se a necessidade de estudos específicos conforme o projeto avance;

4. A criação de um ambiente para testes do método de detecção proposto se efetuará mediante a utilização do mini computador CuBox, cedido pelo PoP-SC/RNP[1], entidade com a qual o graduando possui vínculo;

5. Os resultados dos testes serão submetidos a um processo de comparação e síntese, possibilitando uma análise aprofundada dos dados coletados.

1.5 ESTRUTURA DO TRABALHO

Este trabalho possui a seguinte estrutura de organização:

- O Capítulo 1, denominado "Introdução", apresenta uma visão geral do trabalho;
- No Capítulo 2, intitulado "Conceitos Básicos", são abordados os fundamentos essenciais que sustentam o desenvolvimento deste projeto;
- No Capítulo 3, "Estado da Arte", contém os trabalhos relacionados a esta proposta;
- O Capítulo 4, "Proposta de Desenvolvimento", abrange a fase de elaboração e realização da proposta, incluindo os testes relevantes;
- Por fim, no Capítulo 5, "Considerações Finais", são apresentadas as conclusões finais e as possibilidades para estudos subsequentes.

[1] www.pop-sc.rnp.br

2 CONCEITOS BÁSICOS

Nesta seção, serão introduzidos alguns conceitos básicos relacionados ao trabalho.

2.1 CIBERSEGURANÇA

Cibersegurança, do inglês *cybersecurity*, é tratado como "coisas que são feitas para proteger uma pessoa, organização ou país e suas informações de computador contra crimes ou ataques realizados usando a Internet" (CAMBRIDGE UNIVERSITY PRESS, 2022a). Segundo Kaspersky Lab (2022), define-se cibersegurança como "a prática de defender computadores, servidores, dispositivos móveis, sistemas eletrônicos, redes e dados de ataques maliciosos", onde tal prática pode ser dividida em algumas categorias, tal como:

- **Segurança de rede**, que é a prática de proteger uma rede de computadores contra invasores, os quais podem ser invasores utilizando ataques direcionados ou um malware oportunista;
- **Segurança de aplicativos**, que se concentra em manter o software e os dispositivos livres de ameaças que podem comprometer algum tipo de dado que foi projetado para ser sigiloso;
- **Segurança da informação**, que protege a integridade e privacidade dos dados, tanto no armazenamento quanto no trânsito;
- **Segurança operacional**, que envolve as permissões que os usuários têm em uma rede e os procedimentos que determinam como e onde os dados podem ser armazenados ou compartilhados;
- **Recuperação de desastres e a continuidade dos negócios**, que define as políticas de recuperação e o plano a ser recorrido após certos tipos de desastres causados, por incidentes de segurança cibernética ou qualquer outro evento que cause a perda de operações ou dados;
- **Educação do usuário final**, que visa ensinar às pessoas boas práticas de segurança e estimulá-las a pôr em prática, pois, o fator humano é o mais imprevisível da segurança cibernética;

O trabalho de Humayun et al. (2020), trata a cibersegurança como "conjunto de ferramentas, técnicas, políticas, medidas de segurança, diretrizes de segurança, estratégias de mitigação de riscos, ações, treinamento, boas práticas, garantia de segurança e tecnologias mais recentes que podem ser usadas para proteger o espaço cibernético e os ativos dos usuários". Ao associar tais termos aos dias atuais, assim como em Lezzi, Lazoi e Corallo (2018) que relaciona com a indústria 4.0, demonstra que a falta de segurança pode significar uma grande perda para as companhias, das quais, poucas se dizem preparadas para enfrentar esse tipo de desafio.

2.2 ENGENHARIA SOCIAL E FATOR HUMANO

Um ataque de engenharia social, segundo Syafitri et al. (2022), busca manipular uma vítima atacando seu ponto mais fraco. Segundo o dicionário de Inglês de Oxford, o termo "engenharia social" pode ter dois significados distintos, o primeiro se trata de usar um planejamento centralizado na tentativa de gerenciar uma mudança social e o segundo se trata de persuadir uma pessoa induzindo-a a divulgar informações privilegiadas. Deve-se ressaltar que teve a primeira ocorrência de tal termo foi em 1842 e ainda se encontra nos dias atuais (HATFIELD, 2018). Alguns tipos de engenharia social, além do phishing, segundo Syafitri et al. (2022), são: *pretexting, baiting e ace-to-face interaction*.

Na técnica de *pretexting*, o atacante coleta uma informação pública disponível em *websites*, redes sociais e listas telefônicas, para elaboração de um ataque de comunicação bidirecional, onde o atacante pode oferecer ou pedir algo ao alvo.

A técnica de *baiting* utiliza a curiosidade de membros de uma organização sobre algum item para que conecte tal dispositivo infectado em algum aparelho da organização, assim infectando-o.

A técnica de *ace-to-face interaction* é comumente usada em ataques de engenharia social. Essa técnica explora tirar vantagem de fraquezas psicológicas da vítima, como implorando ajuda ou por acesso a algum dispositivo.

Com isso, considerando que estamos cercados por tecnologia da qual buscamos o uso para melhorar nossas vidas, essa também acaba por deixar-nos mais vulneráveis e acessíveis a sistemas enganosos e à exploração.

Para considerar um sistema seguro, as pessoas responsáveis pelo sistema devem seguir normas e padrões. Dessa forma, considerando que as pessoas podem ter distração, pressão psicológica, estresse ou outras questões, que podem causar o surgimento de fraquezas no sistema e, consequente, um invasor pode explorar tal fraqueza consequente de um fator humano (DESOLDA et al., 2021).

2.3 MALWARE

Malware, em Cambridge University Press (2022b) é definido como "software projetado para danificar as informações nos computadores de outras pessoas e impedir que os computadores funcionem normalmente". No artigo Almeida (2022), um *malware* pode ser classificado como um hardware, firmware ou software que, intencionalmente ou não, possam ser inseridos ou adicionados a um sistema e que possui a capacidade de comprometer sua confiabilidade, integridade ou disponibilidade dos dados, aplicativos ou do sistema operacional da vítima.

Pode se tratar de um software ou hardware projetados intencionalmente para uma causa específica, tal como roubar dados pessoais de um cliente ou de uma rede de computadores. Existe uma variedade de *malware*, como vírus e Trojan, e também alguns tipos de *phishing* que envolvem *malware*, tal como SMiShing, cujo anexo pode instalar um software mal intencio-

nado, como um *rootkit* ou *backdoor* (DESOLDA et al., 2021).

Um dos tipos de *malware* mais notáveis e que está diretamente ligado a ataques de *phishing* e outros ataques de engenharia social é *Ransomware*. Um tipo de *malware* projetado para facilitar atividades como roubo de dados por ser um ataque mais direcionado aos usuários finais. Com a finalidade de reter dados pessoais e liberá-los apenas mediante ao pagamento de um resgate, tornou- se uma forma muito lucrativa para cibercriminosos. O número desse tipo de ataques cresceu exponencialmente nos últimos anos por conta da disponibilidade de ferramentas e tipos de serviços de *ransomware* (RaaS) que facilitam o surgimento de novos atacantes (ALMEIDA, 2022).

2.4 PHISHING

Phishing é um ataque que explora técnicas de engenharia social para realizar um roubo de informações confidenciais (ALEROUD; ZHOU, 2017), estando presente na sociedade desde 1995 (JAMES, 2006). O termo *phishing* se dá pelo fato de que o atacante está tentando pescar, do inglês *fishing*, dados; e o 'ph' é derivado de sofisticado, do inglês *sophisticated*, por conta das técnicas mais sofisticadas que tais atacantes usam para se distinguir da atividade mais simples de pescar (JAMES, 2006).

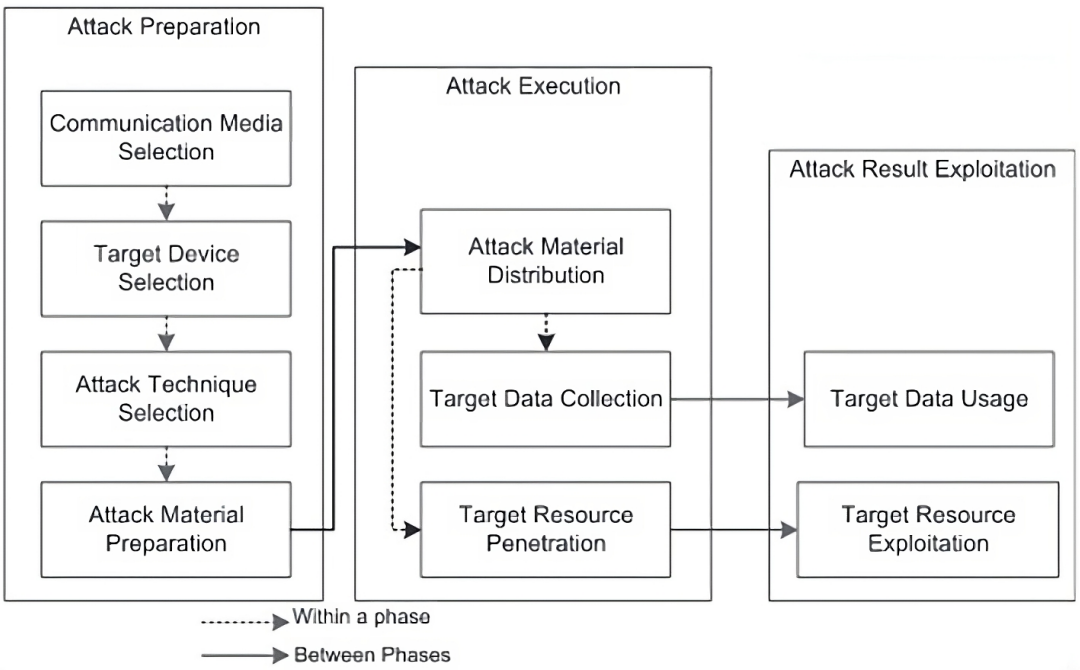

Figura 1 – Fases do processo de um ataque phishing (ALEROUD; ZHOU, 2017)

2.4.1 Processo de elaboração

Nessa seção serão explicadas as fases apresentadas na Figura 1.

Na primeira, é selecionada a mídia que será o veículo do ataque, o dispositivo alvo, a técnica de abordagem e a preparação do material para o ataque. Após isso, na segunda fase, há a execução do ataque, onde se distribui o material contaminado colocando na coleção de dados do alvo ou em seus recursos. Com isso, caso o alvo o acesse, irá ocasionar no fornecimento de dados sensíveis ou relevantes ao atacante. Caso o ataque seja realizado com outros tipos de ataques, como o de *ransomware*, pode acarretar em outros problemas, como a contaminação do dispositivo.

Na Figura 2, pode-se ver alguns exemplos de meios, vetores e a técnica de abordagem para a elaboração de um ataque de *phishing*.

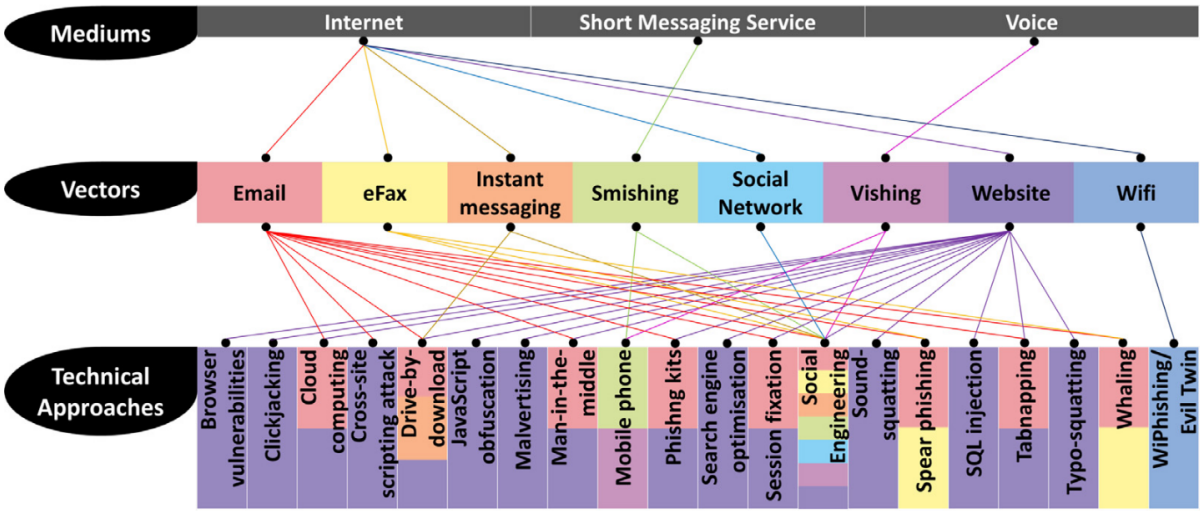

Figura 2 – A interligação entre meio, vetor e abordagem das técnicas de phishing (CHIEW; YONG; TAN, 2018)

2.4.1.1 Meios

Para que um ataque de *phishing* aconteça, é necessário ter um meio para haver a interação entre o atacante e o alvo, e, para que isso aconteça, existem alguns meios mais comuns, que seria o meio da Internet, *Short Messaging Service* (SMS), meios dos quais pessoas utilizam normalmente e que podem ser usados por atacantes para interagir com vítimas (CHIEW; YONG; TAN, 2018).

2.4.1.2 Vetores

Existem vários vetores associados aos meios citados anteriormente que servem para intermediar um ataque por um meio e uma abordagem escolhida, contudo, vetores associados ao meio de Internet são os mais populares entre os ataques de *phishing* (CHIEW; YONG; TAN, 2018). Pode-se destacar os vetores de Email, para Internet, Smishing, para SMS, e Vishing, para serviços de voz.

2.4.1.3 Abordagens Técnicas

Existem várias abordagens técnicas que podem ser usadas em um ou mais vetores para a implementação de um ataque de *phishing*, assim como mostrado na Figura 2. Algumas abordagens altamente relacionadas ao *phishing* são: o *Spear phishing*, um ataque direcionado a um indivíduo, um grupo ou organização, onde se desenvolve um email com conteúdo relevante e que a vítima conhece o remetente, assim, evitando alguma suspeita. O *Whaling*, um tipo de *spear phishing*, o qual tem como alvo pessoas de alto nível executivo e com altos privilégios de acesso na organização (CHIEW; YONG; TAN, 2018).

2.5 BANCOS DE DADOS DE PHISHING

Os banco de dados de *phishing* se tratam de acervos de dados e informações coletadas na Internet sobre *phishing*. Neste trabalho será usado o banco de dados *PhishTank* tal como o *Phishload* e *Openphish*.

O *PhishTank* é um banco de dados colaborativo de *phishing* disponível na Internet. Esse sistema fornece uma API[1] aberta para desenvolvedores. Seus dados são fornecidos em vários formatos e atualizados de hora em hora, assim colaborando com uma aplicação mais rápida e atualizada na detecção de *phishing* (CISCO SYSTEMS, 2022).

O *Openphish* é um banco de dados limitado na sua forma gratuita e com planos para assinatura. Esse sistema fornece dados atualizados de informações, dados e metadados sobre *phishing* e que podem ser usados para detectar e analisar incidentes cibernéticos (OPENPHISH, 2022).

2.6 DOCKER & DOCKER COMPOSE

O *Docker*, um software de código aberto, se destaca como uma plataforma que capacita os desenvolvedores a automatizar o ciclo de vida das aplicações, abrangendo desde sua criação até sua execução em ambientes isolados. Nesse contexto, são fornecidos ambientes virtuais que englobam todos os elementos necessários para garantir a execução consistente e independente das aplicações, desde o estágio de desenvolvimento até a produção. O *Docker* oferece uma solução eficiente para isolar aplicativos e suas dependências, viabilizando a movimentação e replicação descomplicada entre diferentes contextos (DOCKER, 2023).

Por sua vez, o *Docker Compose* é uma ferramenta que simplifica a definição e gestão de aplicações que operam no ambiente *Docker*. Ele possibilita descrever de maneira eficaz a infraestrutura de uma aplicação e suas interações entre os componentes, tornando mais acessíveis os processos de implantação e escalabilidade de aplicações mais complexas. Além disso, o *Docker Compose* também permite a configuração de ambientes de desenvolvimento locais que

[1] APIs são mecanismos que permitem que dois componentes de software se comuniquem usando um conjunto de definições e protocolos (AMAZON WEB SERVICES, 2022).

se assemelham aos ambientes de produção, garantindo uniformidade ao longo do ciclo de vida do desenvolvimento de software (DOCKER, 2023).

2.7 DNS & BIND9

De acordo com a RFC 1034 Group (1987), o DNS (*Domain Name System*) representa um elemento vital da infraestrutura da Internet, projetado para solucionar o desafio de vincular nomes de domínio, que são de fácil compreensão para os seres humanos, como "www.exemplo.com", aos correspondentes endereços IP numéricos, tais como "192.168.0.1". Esse sistema de nomenclatura, caracterizado por sua natureza hierárquica e descentralizada, assume um papel de primordial relevância para a operacionalidade da Internet. Ele estabelece uma estrutura em forma de árvore que organiza de maneira eficiente os domínios e subdomínios, simplificando, assim, o processo de resolução de nomes. Essa abordagem possibilita que os servidores realizem consultas entre si, com o intuito de localizar o endereço IP preciso associado ao domínio desejado.

Por outro lado, o BIND9, um servidor de nomes de domínio (DNS) de código aberto, é amplamente reverenciado por sua notável flexibilidade. Ele permite que um único servidor DNS assuma múltiplas funções, que abrangem desde a atuação como servidor de nomes autoritativo e resolvedor até a capacidade de desempenhar a função de resolvedor de encaminhamento, quando utilizado em sistemas compatíveis. A sólida reputação do BIND9 é fundamentada em sua confiabilidade, robustez e adaptabilidade, tornando-o especialmente adequado para atender às demandas diversificadas de uma ampla gama de implementações de DNS, desde redes locais de pequeno porte até ambientes globais com alto volume de tráfego (CONSORTIUM, 2023).

2.8 ARM

O termo ARM (*Advanced RISC Machines*) refere-se a uma arquitetura de conjunto de instruções (ISA) desenvolvida pela renomada empresa ARM Holdings, atualmente reconhecida como Arm Limited. Essa arquitetura é amplamente difundida entre dispositivos embarcados, smartphones, tablets e uma extensa diversidade de outros sistemas. O notável atributo que distingue os processadores ARM é sua capacidade de proporcionar um desempenho elevado, mantendo simultaneamente um consumo eficiente de energia. Essa característica torna esses processadores especialmente indicados para uma vasta gama de aplicações, que incluem desde dispositivos móveis até sistemas de controle incorporados em uma variedade de setores (ARM, 2014).

3 ESTADO DA ARTE

Nesta seção, serão apresentados os métodos empregados na revisão bibliográfica conduzida neste trabalho, juntamente com uma análise dos principais métodos de detecção propostos no cenário atual de pesquisa. A seção Seção 3.1 engloba uma revisão aprofundada da literatura relacionada ao *phishing* e são destacados estudos relevantes que contribuíram significativamente para a elaboração deste trabalho.

3.1 REVISÃO BIBLIOGRÁFICA SISTEMÁTICA

Para realizar uma revisão sobre *phishing*, foi feito um levantamento das publicações realizadas desde 2019, na plataforma do *google scholar*[1], classificadas de acordo com suas respectivas abordagens.

Levantamento das publicações utilizando o idioma Inglês, usando a configuração de busca "Em qualquer idioma"e com a opção "incluindo citações"marcada:

Tabela 1 – Resultados da revisão sistemática por palavra-chave em inglês.

Palavra-chave	Total
Phishing	28.700
Phishing Detection	20.600
DNS Based Phishing Detection	15.000
Phishing Detection, Cyberattack	17.200
Phishing Detection, Cyberattack, Anti-phishing	2.070
Phishing Detection, Cyberattack, Anti-phishing, DNS Based Phishing Detection	631

Fonte: Elaborada pelo autor em Outubro de 2023.

Levantamento das publicações utilizando o idioma Português, usando a configuração de busca "Pesquisar páginas em Português"e com a opção "incluindo citações"marcada:

Tabela 2 – Resultados da revisão sistemática por palavra-chave em português.

Palavra-chave	Total
Phishing	1.540
Detecção de phishing	663
Detecção de phishing baseada em DNS	143
Detecção de Phishing, Ataque Cibernético	458
Detecção de Phishing, Ataque Cibernético, Antiphishing	40
Detecção de Phishing, Ataque Cibernético, Antiphishing, Detecção de Phishing Baseada em DNS.	20

Fonte: Elaborada pelo autor em Outubro de 2023.

Pode-se notar que há uma diferença de resultado significativa ao se considerar o idioma

[1] https://scholar.google.com.br

da pesquisa. Com isso, foram selecionados alguns trabalhos, com base em suas características, relevância sobre o tema de *phishing* e também a nacionalidade.

3.1.1 Trabalhos relacionados

Nesta subseção, será apresentado alguns dos trabalhos relacionados. Os textos foram selecionados em virtude de suas contribuições referente ao tema *phishing* para a comunidade acadêmica, assim como, sua nacionalidade e autoria. O primeiro, apresentado na Subsubseção 3.1.1.1, é utilizado como base para a proposta de desenvolvimento deste trabalho. O segundo, apresentado na Subsubseção 3.1.1.2, é de autoria de um pesquisador brasileiro renomado na área e apresenta um conteúdo técnico relevante e aprofundado sobre o tema. Por fim, o apresentado na Subsubseção 3.1.1.3, é um trabalho de origem internacional que aborda uma forma de detecção diferente das abordadas nos trabalhos supracitados, além de desenvolver um bom domínio sobre o tema.

3.1.1.1 Heuristic phishing detection based on web crawling and user behaviour monitoring with a deterministic approach for cybersecurity

Em Almeida (2022) propõe um método heurístico para detecção de *phishing* com base em verificação de strings específicas em URLs e mensagens de e-mail. Esse método, juntamente com a monitoração do comportamento do usuário, são capazes de detectar a presença dos principais elementos que têm contato direto com o usuário além de também possibilitar monitorar o comportamento do usuário, podem ser utilizados na detecção de manobras maliciosas em tempo real. Por conseguinte, os resultados demostram que o método é eficaz na detecção de *phishing* e ações do usuário, no entanto, é observado algumas limitações relacionadas ao processamento de páginas complexas. O trabalho Almeida (2022) servirá como base para a proposta de desenvolvimento deste trabalho ao fornecer algoritmos, estratégias e estudos importantes. Deve-se ressaltar que o autor do trabalho anteriormente citado, Rômulo A. C. B. de Almeida, é o coorientador deste trabalho em desenvolvimento.

3.1.1.2 Heuristic-based strategy for Phishing prediction: A survey of URL-based approach

Em Silva, Feitosa e Garcia (2020) trata sobre os comportamentos homográficos [2] comumente presentes em ataques de *phishing* associados a uma determinada marca-alvo, seja na URL ou no conteúdo da página. Monitorou o impacto de aspectos da identidade textual em domiınios, subdomiınios e uso de suas palavras-chave em um ataque de *phishing* e os desafios sobre a proteção da marca em ataques homógrafos. O estudo envolveu uma pesquisa com 12 características, coletadas tanto a partir do próprio estudo quanto de fontes externas, que foram

[2] Palavra que tem a mesma grafia que outra palavra mas tem um significado diferente (CAMBRIDGE UNIVERSITY PRESS, 2022c).

aplicadas a conjuntos diferentes de sites de *phishing* e sites legítimos de 2018. Os resultados sugerem que, embora tenha havido progresso na previsão de *phishing*, algumas características são menos relevantes e podem precisar ser descartadas, enquanto outras parecem mais consistentes em cenários de *phishing* e justificam investigações adicionais.

O trabalho Silva, Feitosa e Garcia (2020) serviu como uma base teórica importante fornecendo pontos sobre *phishing*, referências e formas de encobrir ameaças para o desenvolvimento deste trabalho. Deve-se ressaltar a importância desse trabalho por ser inteiramente em português.

3.1.1.3 NoFish; Total Anti-Phishing Protection System

Em Atimorathanna et al. (2020) é apresentada uma solução chamada "NoFish"para detecção de ataques de *phishing* identificando o site que o usuário está prestes a visitar, incluindo a detecção de logotipos e características importantes usando processamento de imagem e aprendizado de máquina. Além disso, o sistema utiliza algoritmos de classificação, aprendizado de máquina e processamento de linguagem natural para análise de URLs de *phishing* e oferece proteção contra ataques relacionados a DNS. "NoFish"também possui um plugin de cliente de e-mail para o Microsoft Outlook, que detecta e-mails de spam e extrai URLs do corpo do e-mail para análise adicional. A pesquisa avalia o desempenho da solução em várias áreas, incluindo detecção de URLs de *phishing*, detecção de e-mails de *phishing* e proteção contra ataques relacionados a DNS. A solução é projetada para ser uma defesa abrangente contra ataques de *phishing*, com um alto nível de precisão.

3.1.2 Comparativo entre trabalhos

Ao comparar os trabalhos encontrados durante a revisão bibliográfica com o proposto neste projeto, chega-se às seguintes conclusões:

Durante a pesquisa, foram identificadas várias metodologias aplicadas aos sistemas de detecção, como abordagens heurísticas, técnicas de aprendizado de máquina e processamento de imagem. Com isso, demonstrando uma grande variedade de métodos robustos para a detecção de *phishing*. No entanto, no contexto deste projeto, foi escolhido uma abordagem mais simples, com etapas bem definidas e de fácil configuração e implementação.

Os sistemas de detecção disponíveis na literatura demonstram um alto nível de eficácia, sendo capazes de identificar URLs completas em páginas e prever possíveis ações dos usuários. Por exemplo, no estudo de Almeida (2022), alcançou-se uma precisão de 97,66% na detecção de páginas de *phishing*. No entanto, no projeto em questão, a métrica de acerto em URLs é mais difícil de se calcular, uma vez que o método verifica apenas o domínio acessado. A eficácia deste projeto pode variar dependendo da rede em que é implementado, tornando-o suscetível a falsos positivos, como discutido na seção 4.2.4.

Quando consideramos a aplicabilidade deste projeto, ele foi desenvolvido com foco em

ser uma solução residencial. Ele incorpora um servidor DNS remoto, o que permite configurações personalizadas de acordo com as necessidades do usuário, também possui uma abordagem de fácil implantação e tem baixo consumo energético. Portanto, apesar de algumas desvantagens em comparação com outras abordagens, este sistema apresenta uma solução eficaz, simplificada e multifuncional.

4 PROPOSTA DE DESENVOLVIMENTO

Nesta seção serão expostos a arquitetura e as características do desenvolvimento usados para a elaboração de um *software* para detecção de *phishing*.

4.1 ARQUITETURA DA REDE PROPOSTA

Na Figura 3, é apresentada a topologia do sistema proposto. O fluxo de execução da análise de um domínio tem início com a solicitação de resolução DNS por parte de um usuário, representado como *Residential Private Networks*, direcionada ao servidor Bind9 em execução no sistema de virtualização em contêineres, conforme destacado no ponto (1). Após receber essa requisição, o servidor irá fornecer o endereço IP associado ao domínio solicitado, conforme referenciado no ponto (2), ao mesmo tempo em que registra os devidos dados dessa operação, conforme descrito no ponto (3).

Quando um novo registro é inserido no arquivo de log, a aplicação detecta e analisa o texto com o objetivo de extrair o domínio solicitado, conforme referenciado no ponto (4). Em seguida, a aplicação verifica se esse domínio é malicioso ou não, baseando-se em um banco de dados de URLs maliciosas obtido por meio de uma API do *PhishTank*, conforme indicado no ponto (5). Se o domínio não for considerado malicioso, nenhum processo adicional é executado. Entretanto, se for identificado como um domínio malicioso, os dados relativos à solicitação maliciosa são registrados em um novo arquivo de log, como indicado no ponto (6). Além disso, um incidente desse tipo é comunicado por meio de uma notificação por e-mail, conforme descrito no ponto (7).

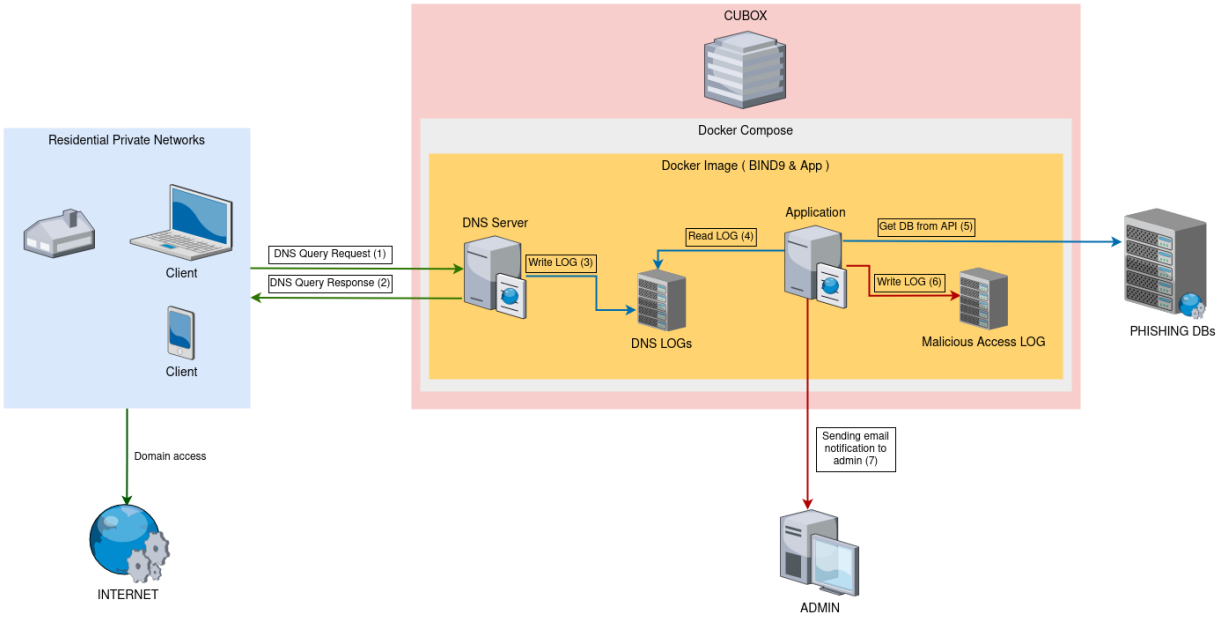

Figura 3 – Topologia do sistema proposto.

A seguir, serão explicitados mais detalhadamente os componentes principais desse sistema.

4.1.1 Componentes Principais

- **CUBOX:**

  Para todos os testes, utilizou-se um minicomputador CuBox, caracterizado por ser uma máquina de hardware robusto e de eficiência energética, capaz de oferecer suporte a uma ampla variedade de aplicações (SOLIDRUN, 2017). Os dados a seguir são as características do CUBOX empregado.

  Vale ressaltar que o hardware que compõe este dispositivo foi desenvolvido com base na arquitetura ARM, o que impactou o desenvolvimento deste projeto, uma vez que o sistema Bind9 não dispõe de uma imagem *Docker* pronta para essa arquitetura. Dessa forma, foi necessário criar uma imagem específica para o projeto.

```
Modelo:
    sr-imx6
CPU:
    Arquitetura: armv7l
    Modelo: Cortex-A9 4 CPUs / 996 MHz
Memoria RAM:
    2011MiB DDR3
Interface Ethernet:
    nome lógico: eth0
    capacidade: 1 Gbit/s
Armazenamento:
    7.4GB Cartão SD
Sistema Operacional:
    Linux sr-imx6 5.10.0-16-armmp
Fonte de alimentação:
    12V 1.5A
    Potência máxima: 15W
Dimensões:
    50mm x 50mm x 50mm
```

Figura 4 – Ficha técnica CuBox.

Figura 5 – Imagem do dispositivo CuBox (SOLIDRUN, 2017).

- **DNS Server & BIND9:**

  Para o sistema proposto, optou-se pelo uso de um servidor DNS devido à sua implementação simples e facilidade de uso para o cliente final. Não requer a instalação de um agente de monitoramento; basta alterar o servidor DNS na máquina a ser analisada; fornece um exemplo de registro de dados que a aplicação utilizará para fins de verificação. No entanto, considerando que os dados de entrada são provenientes de registros de servidor (logs), é possível adaptar facilmente o sistema proposto para usar entradas de outro tipo de serviço que registre URLs acessadas pelos usuários, como registros de logs de acessos gerado por um sistema de firewall.

  Para essa função de servidor DNS, foi escolhido o BIND9, devido à sua leveza, compatibilidade com a imagem Alpine para *Docker*, fácil de implementação e por possuir a funcionalidade de gerar registros de logs para as requisições.

- **Application:**

  A aplicação proposta encarregada da verificação dos domínios foi desenvolvida em Python (PYTHON SOFTWARE FOUNDATION, 2022). Ela é encarregada de baixar a base de dados do *PhishingTank* que será utilizada para a verificação de domínios maliciosos. Neste trabalho será usado o banco de dados *PhishTank* por conta de já possuir o fornecimento de uma conta para utilização de sua API, porém há outros bancos de dados que podem ser utilizados. Também é responsável por monitorar os registros de logs de domínios gerados pelo servidor DNS, a fim de rastrear os domínios acessados pelos usuários. Dessa forma, ao atualizar a base de dados e analisar os registros de log DNS, a aplicação busca o domínio registrado nos logs de DNS na base de dados. Se o domínio for encontrado, ele é registrado em um novo arquivo de log de acesso malicioso (*Malicious Access LOG*) contendo a consulta DNS realizada e o domínio malicioso acessado. Posteriormente, um e-mail de notificação é enviado para um usuário predefinido.

- **Script de Inicialização:**

  Um *Shell script* foi desenvolvido com o propósito de configurar e iniciar a aplicação Python, juntamente com o servidor BIND, fornecendo previamente todos os parâmetros necessários para garantir sua inicialização adequada.

- **Docker e Docker Compose:**

  Os serviços *Docker* e *Docker Compose* desempenham um papel fundamental na configuração e inicialização adequada de todos esses serviços mencionados anteriormente. Foi necessário montar uma imagem *Docker* por conta que a imagem do serviço BIND9 disponível no dockerhub[1] não é compatível com a arquitetura ARM.

  O *Dockerfile* apresenta as seguintes características de configuração do sistema:

[1] https://hub.docker.com

- Utilização de uma imagem Alpine como base, devido à compatibilidade com os softwares que serão utilizados, sua leveza e capacidade de executar em arquitetura ARMv7.
- Instalação de todas as dependências básicas, incluindo a atualização do fuso horário, Python e, entre outros, a instalação do serviço de DNS.
- Estabelecimento de um volume contendo os arquivos necessários para o projeto, a serem copiados para o sistema.
- Foi definido a variável de *timezone* como 'TZ' para atualizar o horário do sistema. Para este projeto está sendo utilizado o zona de America/Sao_Paulo por comodidade na aferição dos registros.

Para facilitar a inicialização e montagem desse sistema, a configuração é realizada por meio de um arquivo de configuração do *Docker Compose*, que possui as seguintes características:

- Definição de volumes para a inicialização.
- Especificação da execução de um *Shell script* responsável por iniciar as aplicações.

4.1.2 Coleta de Dados

O sistema Python está constantemente monitorando as entradas do arquivo de log do servidor DNS. Ao encontrar uma entrada, no formato mostrado na Figura 6, ele realiza uma análise minuciosa recortando o conteúdo após 'query:' do qual é referente ao domínio acessado. O resultado desse recorte é então encaminhado para o método de consulta no banco de dados. Esse já tem previamente carregado com um conjunto de dados relacionados a *phishing* e, neste momento, realiza uma pesquisa para verificar a existência desse domínio. Se o domínio for identificado como pertencente a uma URL maliciosa, por exemplo 'dominio.com' que parte de 'https://dominio.com/sub_caminho_malicioso', o domínio juntamente com o registro de log DNS é arquivado em um novo registro referente a domínios maliciosos acessados, seguindo a Figura 7. Após essa etapa, é gerado e enviado um e-mail para uma conta registrada, assim como mostrado na Figura 8, a fim de informar sobre o incidente de segurança que ocorreu.

```
------------------------------------------------------------
DNS query:01-Sep-2023 21:47:42.071 queries: info: client
@0xb5a122d5 192.168.3.9#60893 (google.com):
query: google.com IN A +E(0)K (142.250.219.238)
------------------------------------------------------------
```

Figura 6 – Exemplo de log DNS.

```
------------------------------------------------------------
DNS query:01-Sep-2023 21:47:42.071 queries: info: client
@0xb5a122d5 192.168.3.9#60893 (malicious.com.br):
query: malicious.com.br IN A +E(0)K (0.0.0.0),
Phishing URL: malicious.com.br/olde/
------------------------------------------------------------
```

Figura 7 – Exemplo de log malicioso.

```
------------------------------------------------------------
Subject: Notificacao de acesso a URL maliciosa.

from: sender@email.com
to: receiver@email.com

Content:
Notificacao de acesso a URL maliciosa.
Query: 01-Sep-2023 21:47:42.071 queries: info: client
@0xb5a122d5 192.168.3.9#60893 (malicious.com.br):
query: malicious.com.br IN A +E(0)K (0.0.0.0)
------------------------------------------------------------
```

Figura 8 – Exemplo de notificação de e-mail.

4.1.3 Desenvolvimento do Sistema de Análise de Phishing

Para a implementação do sistema proposto, foi necessário seguir os passos abaixo.

Estrutura da aplicação Python é composta por três arquivos Python principais, um arquivo CSV contendo os domínios maliciosos e dois arquivos de registro, assim como mostra a Figura 9. O arquivo A.1 desempenha um papel central, responsável por carregar o banco de dados armazenado em *online-valid_1.csv* (arquivo baixado por meio da API do PhishTank), assim como demonstrado na Figura 10. Ele monitora e analisa os registros gerados pelo servidor DNS em busca de domínios solicitados, encaminhando qualquer domínio encontrado para o método de verificação em A.2. No caso de um domínio ser identificado como malicioso, os dados são registrados em *phishing_queries.log* no padrão demonstrado na Figura 7 e um método definido em A.3 é invocado para enviar uma notificação por email.

```
------------------------------------------------------------
-- app
   |-- banco_dados_phishing
   |   `-- online-valid_1.csv
   |-- phishing_logs
   |   |-- log_counter.log
   |   `-- phishing_queries.log
   |-- api_conect.py
   |-- log_monitor.py
   `-- send_email.py
------------------------------------------------------------
```

Figura 9 – Topologia de arquivos da aplicação.

```
------------------------------------------------------------
phish_id,url,phish_detail_url,submission_time,verified,verification_time,
online,target

8309220,https://accont-cofirmed-color...,http://www.phishtank.com/
phish_detail.php?phish_id=8309220,2023-09-23T17:01:38+00:00,yes,2023-09
-23T17:13:14+00:00,yes,Facebook

8309219,https://home15-kal3mpongs909...,http://www.phishtank.com/
phish_detail.php?phish_id=8309219,2023-09-23T17:01:19+00:00,yes,2023-09
-23T17:13:14+00:00,yes,Facebook

8309218,https://accont-cofirmed-color...,http://www.phishtank.com/
phish_detail.php?phish_id=8309218,2023-09-23T17:00:57+00:00,yes,2023-09
-23T17:04:08+00:00,yes,Facebook

8309217,https://dhdhaggee.wixsite...,http://www.phishtank.com/
phish_detail.php?phish_id=8309217,2023-09-23T16:59:58+00:00,yes,2023-09
-23T17:04:08+00:00,yes,Other

...
------------------------------------------------------------
```

Figura 10 – online-valid_1.csv

Também possui um contador de número de logs, como exemplificado na Figura 11, que é executado durante um período de atividade do programa para se ter ciência da quantidade de requisições que foram analisadas. O contador é reiniciado toda vez que o sistema é iniciado.

```
------------------------------------------------------------
time: 2023-10-26 21:08:16.375445, Number of logs analyzed: 1
time: 2023-10-26 21:08:17.582027, Number of logs analyzed: 2
time: 2023-10-26 21:08:18.783774, Number of logs analyzed: 3
time: 2023-10-26 21:08:23.001434, Number of logs analyzed: 4
time: 2023-10-26 21:08:24.200980, Number of logs analyzed: 5
time: 2023-10-26 21:08:25.404976, Number of logs analyzed: 6
time: 2023-10-26 21:15:04.095818, Number of logs analyzed: 1
time: 2023-10-26 21:15:05.319932, Number of logs analyzed: 2
time: 2023-10-26 21:15:06.524617, Number of logs analyzed: 3
------------------------------------------------------------
```

Figura 11 – log_counter.log

named.conf definido em A.4, se trata da configuração usada no servidor Bind9. Foram definidos os servidores DNS que o servidor local irá buscar quando não for capaz de resolver o domínio pedido, definido os parâmetros para que o servidor gere logs de todas as resoluções que ele responder e também foi adicionado uma regra para que o servidor apenas resolva requisições de IPs dentro de uma rede especificada.

entrypoint.sh definido em A.7, se trata de um *script* com as definições necessárias para a devida configuração e inicialização das aplicações.

Docker e Docker Compose tendo os softwares do *Dockerfile* e *Docker Compose* instalados segundo sua documentação, o A.5 foi configurado da seguinte forma: foi escolhida a imagem Alpine devido à sua compatibilidade com o projeto; em seguida, foram definidas as dependências necessárias para o projeto, incluindo o Python para a execução da aplicação de verificação de logs e a instalação das bibliotecas necessárias; o servidor DNS BIND foi instalado para servir como servidor DNS, o tzdata foi configurado para ajustar o fuso horário do sistema; a porta necessária para o servidor DNS foi configurada para exposição; foi definida a cópia dos arquivos essenciais do projeto da pasta local para o sistema que será iniciado pelo software de virtualização em contêineres; por fim, foi estabelecido o arquivo de inicialização das aplicações.

Com isso, para a inicialização do A.5, foi elaborado um A.6 no qual são especificados diversos parâmetros essenciais para o correto funcionamento desse sistema. Entre eles estão: a configuração dos volumes, o mapeamento das portas e os protocolos destinados ao servidor DNS e, adicionalmente, são definidas as políticas de reinicialização do sistema, os métodos de montagem e os locais de montagem da imagem. Essas configurações garantem a execução adequada e eficiente do projeto em um ambiente de virtualização *Docker*. Entretanto, geralmente a porta 53, da qual será utilizada pelo servidor DNS, já está em uso internamente pelo serviço ‘resolved’ em sistemas operacionais Linux[2], para isso pode ser necessário alterar o arquivo localizado em ’/etc/systemd/resolved.conf’ adicionando a seguinte alteração ’DNSStubListener=no’ e então reiniciar o serviço com ’sudo systemctl restart systemd-resolved.service’, assim a porta deixará de ser usada e estará livre para uso.

[2] https://www.linux.org/

4.2 RESULTADOS

Nesta seção serão expostos os resultados obtidos e as análises desses dados.

4.2.1 Taxas de Detecção

Os testes presentes nesta seção compreendem o escopo de eficiência na detecção de domínios em relação aos dados presentes no banco, considerando a verificação da taxa de falsos positivos detectados.

Para gerar esses dados foram efetuados dois testes sintéticos, um com exatas 100 requisições e outro com 1000 requisições, em que ambos os testes possuem 10% de domínios maliciosos e um número variável de domínios que possuem relatos maliciosos porém utilizam domínios oficiais da Google ou Microsoft para hospedar alguma forma de *phishing*.

4.2.1.1 Teste com 100 requisições

Na Figura 12 são apresentados os resultados provenientes do teste consistindo em 100 requisições. Observa-se que 2% das requisições feitas a domínios associados ao Google e a Microsoft foram erroneamente identificadas como maliciosas por conta que, por mais que sejam dominios muito famosos e utilizados na internet, possui alguns relatos de *phishing* hospedados neles, assim ressaltando a necessidade de uma verificação extra sobre o quão malicioso cada domínio é. Por outro lado, é importante ressaltar que todas as requisições a domínios autenticamente maliciosos, que representam 10% do total, foram identificadas com precisão (sinalizados com '<phishing'>).

```
------------------------------------------------------------
    <phishing>attdkjdposervice12.weeblysite.com
    <phishing>www.smbec-caserd.co.jp.y725.top
    <phishing>messagerie68.godaddysites.com
    <phishing>ipfs.eth.aragon.network
    <falso positivo>forms.office.com
    <phishing>dell-com-viewer.firebaseapp.com
    <phishing>rv0ld9.webwave.dev
    <phishing>hsbc-deviceapproval.com
    <falso positivo>docs.google.com
    <phishing>ANY.aeonkv.com
    <phishing>banrraurls.web.app
    <phishing>janusz.denisbiernacki.pl
------------------------------------------------------------
```

Figura 12 – Teste de 100 requisições.

O sistema, ao verificar os domínios *docs.google.com* e *forms.office.com*, acabou relacionando as seguintes URLs maliciosas hospedadas de forma legítima nesses domínios, como apresentado na Figura 13.

```
------------------------------------------------------------
DNS query:11-Nov-2023 16:09:47.637 queries: info: client @0xb5a072c4 192.168.
3.9#37933 (bafkreib5ewve2wdiuqe3xxrkkwxxayaq74ozu64rncc22elezichfimjgi.ipfs.
cf-ipfs.com): query: bafkreib5ewve2wdiuqe3xxrkkwxxayaq74ozu64rncc22elezichfim
jgi.ipfs.cf-ipfs.com IN A +E(0)K (172.23.0.2), Phishing URL:https://
bafkreib5ewve2wdiuqe3xxrkkwxxayaq74ozu64rncc22elezichfimjgi.ipfs.cf-ipfs.com

DNS query:11-Nov-2023 16:09:48.637 queries: info: client @0xb5a55344 192.168.
3.9#50599 (docs.google.com): query: docs.google.com IN A +E(0)K (172.18.0.2),
Phishing URL:https://docs.google.com/presentation/d/e/2PACX-1vRwu-JgwOMxtEHBw
qWMnPtQiH7UH4AkCnwFa5YxUU0pipUnz3oXNc42wy5P3YkIWlB4nT5aZlQYOsBU/pub
------------------------------------------------------------
```

Figura 13 – Registro de Log de domínios maliciosos.

Com isso, é necessário que o gerente da rede verifique os logs e trate juntamente ao usuário da máquina que efetuou o acesso, para prevenir possíveis problemas relacionados a *phishing*.

4.2.1.2 Teste com 1000 requisições

Na Figura 14, é apresentado os resultados decorrentes do composto por 1000 requisições. Pode-se observar um cenário semelhante ao teste anterior, no qual domínios oficiais, como do Google, Microsoft e GitHub, são suscetíveis a falsos positivos devido a sua alta frequência de acesso.

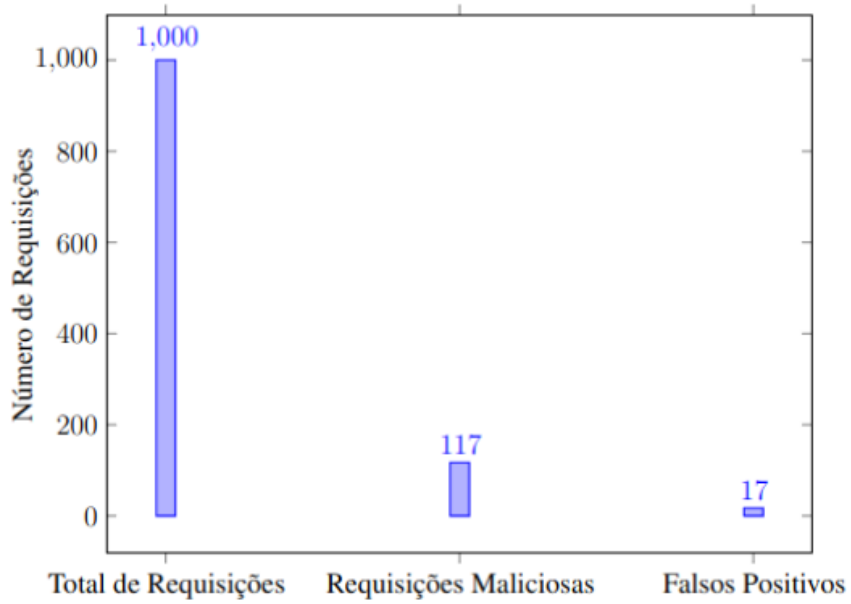

Figura 14 – Teste de 1000 requisições.

4.2.2 Teste de wildcard de domínios

Considerando que vários subdomínios, por exemplo 'docs.google.com', podem ser mapeados para um único domínio utilizando *wildcard* (CLOUDNS, 2023), por exemplo 'google.com'. Com isso, este teste irá analisar a detecção de subdomínios e domínios maliciosos.

No exemplo abaixo, podemos observar a detecção de um subdomínio malicioso presente no banco de dados, denominado 'gateway.ipfs.io'. Quando esse subdomínio é consultado (Figura 15 - 1), o sistema o identifica como malicioso (Figura 15 - 4). Contudo, ao realizar a busca pelo domínio 'ipfs.io' (Figura 15 - 2), ele também é reconhecido como malicioso (Figura 15 - 5), uma vez que está contido na consulta original do banco de dados, que é 'gateway.ipfs.io'. Por outro lado, ao realizar uma consulta que não está presente nas entradas originais do banco de dados, como no caso da *query* 'asd.gateway.ipfs.io' (Figura 15 - 3), ela não é reconhecida como maliciosa. Com isso, pode-se ressaltar que a base de dados utilizada não é tão adequada para a verificação de domínios maliciosos, uma vez que um domínio sendo malicioso todo subdomínio dele também deveria ser considerado malicioso, entretanto, o contrario dessa afirmação pode não ser verdade por conta que um atacante pode utilizar um domínio do qual conseguiu acesso e que não é malicioso para gerar um subdomínio malicioso.

```
------------------------------------------------------------
Registro de domínios do servidor DNS:
1:
08-Nov-2023 20:42:03.038 queries: info: client @0xb5a5e3f4 192.168.3.9#33311
(gateway.ipfs.io): query: gateway.ipfs.io IN A +E(0)K (172.18.0.2)
2:
08-Nov-2023 20:42:06.707 queries: info: client @0xb5a55344 192.168.3.9#49379
(ipfs.io): query: ipfs.io IN A +E(0)K (172.18.0.2)
3:
08-Nov-2023 20:42:11.187 queries: info: client @0xb5a54424 192.168.3.9#59885
(asd.gateway.ipfs.io): query: asd.gateway.ipfs.io IN A +E(0)K (172.18.0.2)

------------------------------------------------------------
Registro de domínios maliciosos detectados pelo sistema:
4:
DNS query:08-Nov-2023 20:42:03.038 queries: info: client @0xb5a5e3f4 192.168.3.9#33311
(gateway.ipfs.io): query: gateway.ipfs.io IN A +E(0)K (172.18.0.2), Phishing URL:
https://gateway.ipfs.io/ipfs/bafkreid4ovxck26zybwzumxngpchys4p7omdcwcz5feptm3xry5tkctp2i
5:
DNS query:08-Nov-2023 20:42:06.707 queries: info: client @0xb5a55344 192.168.3.9#49379
(ipfs.io): query: ipfs.io IN A +E(0)K (172.18.0.2), Phishing URL:https://gateway.ipfs.io
/ipfs/bafkreid4ovxck26zybwzumxngpchys4p7omdcwcz5feptm3xry5tkctp2i


------------------------------------------------------------
```

Figura 15 – Teste de wildcard.

4.2.3 Comparação de execução em arquitetura ARM e x86_64

Neste experimento, será realizado uma análise da compatibilidade do sistema desenvolvido para execução no ambiente de virtualização em contêineres *Docker*. O objetivo é comparar o consumo de recursos em diferentes ambientes, destacando as divergências identificadas. Com isso, proporcionando uma compreensão mais abrangente do desempenho do sistema desenvolvido.

Características da máquina virtual que será utilizada neste teste:

```
Sistema de virtualização: virtualbox(6.1.38_Ubuntu r153438)
CPU:
    Arquitetura: x86_64
    Modelo: Intel i5 9300h 4 CPUs / 4100 MHz
Memoria RAM:
    4000MiB DDR4
Interface Ethernet:
    Adaptador modo Bridge
    capacidade: 1 Gbit/s
Armazenamento:
    20GB ssd
Sistema Operacional:
    ubuntu-23.04-desktop-amd64
Fonte de alimentação do host:
    Voltagem de Entrada: 100 - 240V
    Potência máxima: 130W
```

Figura 16 – Ficha técnica máquina virtual x86_64.

A parte de inicialização do sistema desenvolvido na máquina virtual foi o mesmo utilizado no mini computador do qual foi utilizado para o desenvolvimento.

Abaixo está o consumo de recursos pelo sistema de virtualização em contêineres em ambas as maquinas de teste. Nota-se que o uso de recursos em ambos é bem semelhante.

```
CONTAINER ID   NAME                                CPU %     MEM USAGE / LIMIT     MEM %     NET I/O           BLOCK I/O         PIDS
21ee60b308eb   phishing-detection-and-dns-server   0.07%     18.89MiB / 1.965GiB   0.94%     96.9kB / 55.5kB   22.6MB / 172kB    13
```

Figura 17 – Uso de recursos - CuBox.

```
CONTAINER ID   NAME                                CPU %     MEM USAGE / LIMIT     MEM %     NET I/O         BLOCK I/O         PIDS
21ee60b308eb   phishing-detection-and-dns-server   102.04%   20.64MiB / 1.965GiB   1.03%     561kB / 277kB   22.6MB / 369kB    15
```

Figura 18 – Uso de recursos em teste - CuBox.

Ao rodar o teste descrito em 4.2.1.2 em ambas as máquinas e ao mesmo tempo, então comparar o uso de recuso em ambas, nota-se que também é bem semelhante, assim como demonstrado nas Figuras 18 e Figura 21.

```
CONTAINER ID   NAME                                CPU %     MEM USAGE / LIMIT     MEM %     NET I/O       BLOCK I/O         PIDS
c5d785528c19   phishing-detection-and-dns-server   0.01%     22.99MiB / 3.815GiB   0.59%     18kB / 886B   36.5MB / 45.1kB   11
```

Figura 19 – Uso de recursos - Máquina Virtual.

```
CONTAINER ID   NAME                                CPU %     MEM USAGE / LIMIT     MEM %     NET I/O       BLOCK I/O         PIDS
c5d785528c19   phishing-detection-and-dns-server   103.31%   35.42MiB / 3.815GiB   0.91%     584kB / 248kB   36.5MB / 537kB   11
```

Figura 20 – Uso de recursos em teste - Máquina Virtual.

Sobre o tempo de resposta, marcado como (2) na Figura 5, não se obteve diferenças significativas entre ambas a implementação em arquitetura ARM e X86_64 por conta da topologia da rede utilizada, da qual as conexões entre os pontos foram os mesmos, resultando no mesmo tempo de resposta (RTT).

Em ambos o tempo de resposta do DNS foi ’Query time: 0 msec’ ressalva que a primeira requisição do domínio sempre tem um valor mais elevado por conta que o servidor DNS ter que descobrir o destino do domínio, após isso ele já fica em cache.

Também foi capturado os dados de temperaturas e uso de CPU em ambas as máquinas. Vale ressaltar que, como são dados de uso da máquina em um todo, se trata de uma visualização do sistema operacional como um topo, com diversos serviços rodando, impactando no desempenho.

Segue abaixo o uso de processamento da máquina virtual durante o teste.

Figura 21 – Uso de recursos em teste - Mini PC.

Dados de temperatura do mini pc:

```
imx\_thermal\_zone-virtual-0
Adapter: Virtual device
temp1:        +43.0 C  (crit = +100.0 C)
```

Figura 22 – Temperatura do mini pc.

Segue abaixo o uso de processamento da máquina virtual durante o teste.

A máquina virtual não possui sensor de temperatura, porém seu sistema hospedeiro sim. Segue a temperatura do sistema hospedeiro da máquina virtual durante o teste:

Figura 23 – Uso de recursos em teste - Máquina virtual.

```
coretemp-isa-0000
Adapter: ISA adapter
Package id 0:  +53.0°C  (high = +100.0°C, crit = +100.0°C)
Core 0:        +52.0°C  (high = +100.0°C, crit = +100.0°C)
Core 1:        +52.0°C  (high = +100.0°C, crit = +100.0°C)
Core 2:        +53.0°C  (high = +100.0°C, crit = +100.0°C)
Core 3:        +53.0°C  (high = +100.0°C, crit = +100.0°C)\\
```

Figura 24 – Temperatura hospedeiro da máquina virtual.

A partir da análise apresentada anteriormente, observa-se que, independentemente da arquitetura e dos recursos disponíveis na máquina, o sistema proposto demonstra uma leveza notável. A execução eficiente requer aproximadamente 1 núcleo de 900 MHz, 20 MiB de memória RAM e uma interface de rede com 100 Mbps. Esses requisitos são suficientes para suportar uma topologia de pequeno porte, destacando a eficácia do sistema mesmo em ambientes com recursos mais modestos.

4.2.4 Problemas

Os problemas encontrados durante todo o processo de desenvolvimento e testes deste projeto foram:

Por ser um sistema de detecção de domínios maliciosos e para isso foi necessário utilizar um banco de dados de URLs maliciosas, abre uma margem de erro nesta verificação por conta que se um domínio pode possuir várias URLs saudáveis e uma delas maliciosa isso não o torna um domínio malicioso, mas sim diminui o nível de confiança nesse domínio. Com isso, nota-se a necessidade de alteração do banco de dados de amostras maliciosas e a categorização dos domínios.

Por possuir uma infraestrutura limitada, foi necessário efetuar testes sintéticos obtendo, dessa forma, uma hipótese de comportamento em vida real. Com isso, seria de maior benefício analítico para os testes se eles fossem feitos em uma infraestrutura devidamente preparada para esse tipo de sistema.

5 CONSIDERAÇÕES FINAIS

Nesta seção, serão introduzidos as conclusões sobre este trabalho e o apontado possíveis trabalhos futuros.

5.1 CONCLUSÃO

O *phishing* é uma ameaça persistente na sociedade e que necessita de constante pesquisa, acerca dos métodos de detecção e mitigação, por conta da evolução e abrangência dos métodos de implementação e execução desse tipo de ataque.

O *phishing* representa uma ameaça significativa à integridade das informações, sendo crucial prevenir o vazamento de dados sensíveis. Com isso, este projeto possibilita e colabora com o desenvolvimento de soluções mais robustas com o objetivo de fortalecer a proteção da informação, assim como delineado na LGPD e na GDPR. Assegurando, dessa forma, a confidencialidade dos dados pessoais dos cidadãos, contribuindo para um ambiente digital mais seguro e protegido.

Neste trabalho foi realizada uma demonstração dos procedimentos e dos componentes empregados na criação de uma aplicação dedicada à detecção de domínios associados a ataques de *phishing*, ao mesmo tempo que oferece funcionalidades para a resolução de solicitações DNS.

O projeto chama a atenção por sua notória simplicidade, contudo, não deixa de apresentar uma boa taxas de detecção de domínios maliciosos. Quando comparado a outras abordagens, ainda que possa ser percebido como menos robusto em termos de complexidade, destaca-se significativamente por sua facilidade de implementação.

O escopo primordial deste estudo é a contribuição para a área de pesquisa em segurança cibernética, com o propósito de mitigar a ameaça contínua que acomete nossa sociedade. Esse projeto se direciona especialmente a ambientes residenciais e se caracteriza pelo uso de dispositivos de baixo consumo energético, o que o torna particularmente adequado a esse contexto.

É crucial observar que devido à natureza desta implementação, a qual se baseia nos domínios dos sites como ponto de partida, existe a possibilidade de ocorrer uma taxa de falsos positivos, como ilustrado na Seção 4.2. Diante desse cenário, recai sobre o administrador de rede a responsabilidade de lidar com esse tipo de incidente, demandando uma abordagem proativa e eficiente na resolução desses problemas. Portanto, é essencial que os profissionais envolvidos estejam preparados para lidar com essas situações com destreza e discernimento, a fim de manter a integridade e a segurança da rede.

Por fim, deve-se mencionar que durante o desenvolvimento desse projeto, foi desenvolvido dois artigos acadêmicos dos quais foram publicados na revista *Communications and Innovations Gazzete* (ComIng) (https://periodicos.ufsm.br/coming/index) e estão listado no Apêndice B e no Apêndice C, assim como o link para cada um deles.

5.2 TRABALHOS FUTUROS

Como aprimoramento para investigações subsequentes, é relevante destacar a questão das verificações de domínios, que podem resultar em falsos positivos. Uma abordagem recomendável é a incorporação de uma lista de domínios a serem excluídos das verificações, reduzindo a ocorrência desses falsos positivos. Também pode ser incorporada a essa solução a criação de uma forma de apresentação em *dashboards* dos domínios maliciosos encontrados suas principais ocorrências para, assim, possibilitar uma forma visual de análise.

Outra consideração benéfica é a implementação de um sistema de gerenciamento de banco de dados, a fim de otimizar o desempenho e a robustez do sistema. Tal ação pode contribuir significativamente para a eficiência e a confiabilidade do sistema, assegurando que ele funcione de maneira eficaz.

Um outro possível trabalho futuro, é a utilização do arquivo de log de saída em análise de confiança sobre os domínios, elaboração de listas de bloqueio e suporte em projetos e comunidades que utilizam tais tipos de dados com finalidade de colaborar e aumentar a segurança cibernética.

Além disso, um passo importante seria a evolução dos testes realizados, tornando-os compatíveis com uma ampla variedade de dispositivos de baixo consumo energético, garantindo que a solução é aplicável em diferentes ambientes domésticos. Essa expansão de compatibilidade ampliaria o alcance e a usabilidade do sistema, tornando-o uma solução mais versátil e adaptável às diversas necessidades do usuário. Também seria de grande valor a execução de testes em ambientes com maior fluxo de uso para análise do comportamento e testes utilizando outras fontes de bancos de dados de *phishing*.

Por fim, um outro possível trabalho futuro seria a o desenvolvimento de uma solução para mitigação em tempo real da requisição DNS ou até mesmo que notifique o usuário antes da resposta da resolução DNS requisitada pelo usuário, ponto (2) na Figura 3. Dessa forma, evoluindo este projeto de um sistema de detecção para um sistema de prevenção de *phishing*.

REFERÊNCIAS

ABROSHAN, H. et al. Covid-19 and phishing: Effects of human emotions, behavior, and demographics on the success of phishing attempts during the pandemic. **IEEE Access**, v. 9, p. 121916–121929, 2021.

ALEROUD, A.; ZHOU, L. Phishing environments, techniques, and countermeasures: A survey. **Computers & Security**, v. 68, p. 160–196, 2017. ISSN 0167-4048. Disponível em: https://www.sciencedirect.com/science/article/pii/S0167404817300810. Acesso em: out. 2023.

ALMEIDA, R. A. O. C. B. de. Heuristic phishing detection based on web crawling and user behaviour monitoring with a deterministic approach for cybersecurity. 2022. Disponível em: https://repositorio.ufsc.br/handle/123456789/234735. Acesso em: out. 2023.

AMAZON WEB SERVICES. **O que é uma API?** 2022. Disponível em: https://aws.amazon.com/pt/what-is/api/. Acesso em: out. 2023.

APWG. **PHISHING ACTIVITY TRENDS REPORT 4th Quarter 2022**. apwg.org, 2022. Disponível em: https://docs.apwg.org/reports/apwg_trends_report_q4_2022.pdf?_gl=1*160goqd*_ga*NjM5MzkzMzg1LjE2OTM3NjUzNzA.*_ga_55RF0RHXSR*MTY5ODUyNjAyMy4yLjEuMTY5ODUyNjA1NC4wLjAuMA.._ga=2.17019580.465536221.1698526024-639393385.1693765370. Acesso em: out. 2023.

ARM, L. **Arm architecture reference manual, armv7-a and armv7-r edition**. 2014. Disponível em: https://developer.arm.com/documentation/ddi0406/latest/. Acessado em Setembro de 2023.

ATIMORATHANNA, D. N. et al. Nofish; total anti-phishing protection system. In: **2020 2nd International Conference on Advancements in Computing (ICAC)**. [S.l.: s.n.], 2020. v. 1, p. 470–475.

BRASIL. Lei nº 13.709, de 14 de agosto de 2018. lei geral de proteção de dados pessoais (lgpd). **Diário Oficial da União, Brasília, DF, 14 ago. 2018.**, Brasília, 2018. Disponível em: https://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/l13709.htm. Acesso em: out. 2023.

CAMBRIDGE UNIVERSITY PRESS. **Cybersecurity**. 2022. Disponível em: https://dictionary.cambridge.org/pt/dicionario/ingles/cybersecurity. Acesso em: out. 2023.

CAMBRIDGE UNIVERSITY PRESS. **Malware**. 2022. Disponível em: https://dictionary.cambridge.org/pt/dicionario/ingles/malware. Acesso em: out. 2023.

CAMBRIDGE UNIVERSITY PRESS. **Meaning of homograph in English**. 2022. Disponível em: https://dictionary.cambridge.org/dictionary/english/homograph. Acesso em: out. 2023.

CHIEW, K. L.; YONG, K. S. C.; TAN, C. L. A survey of phishing attacks: Their types, vectors and technical approaches. **Expert Systems with Applications**, v. 106, p. 1–20, 2018. ISSN 0957-4174. Disponível em: https://www.sciencedirect.com/science/article/pii/S0957417418302070. Acesso em: out. 2023.

CISCO SYSTEMS. **PhishTank**. 2022. Disponível em: https://phishtank.org/. Acesso em: out. 2023.

CLOUDNS. **What is Wildcard DNS Record?** 2023. Disponível em: https://www.cloudns.net/wiki/article/190/. Acesso em: nov. 2023.

CONSORTIUM, I. I. S. **BIND 9 Administrator Reference Manual**. 2023. Disponível em: https://bind9.readthedocs.io/en/v9.18.14/chapter1.html. Acessado em Setembro de 2023.

DESOLDA, G. et al. Human factors in phishing attacks: A systematic literature review. Association for Computing Machinery, New York, NY, USA, v. 54, n. 8, 2021. ISSN 0360-0300. Disponível em: https://doi.org/10.1145/3469886. Acesso em: out. 2023.

DOCKER, I. **Docker overview**. 2023. Disponível em: https://docs.docker.com/get-started/overview/. Acessado em Setembro de 2023.

GROUP, I. N. W. **DOMAIN NAMES - CONCEPTS AND FACILITIES**. 1987. Disponível em: https://www.ietf.org/rfc/rfc1034.txt. Acessado em Setembro de 2023.

HATFIELD, J. M. Social engineering in cybersecurity: The evolution of a concept. **Computers & Security**, v. 73, p. 102–113, 2018. ISSN 0167-4048. Disponível em: https://www.sciencedirect.com/science/article/pii/S0167404817302249. Acesso em: out. 2023.

HUMAYUN, M. et al. Cyber security threats and vulnerabilities: A systematic mapping study. **Arabian Journal for Science and Engineering**, v. 45, 2020. Disponível em: https://rdcu.be/cW2Tg. Acesso em: out. 2023.

JAMES, L. Chapter 1 - banking on phishing. In: JAMES, L. (Ed.). **Phishing Exposed**. Burlington: Syngress, 2006. p. 1–35. ISBN 978-1-59749-030-6. Disponível em: https://www.sciencedirect.com/science/article/pii/B9781597490306500064. Acesso em: out. 2023.

KASPERSKY LAB. **What is Cyber Security?** 2022. Disponível em: https://www.kaspersky.com/resource-center/definitions/what-is-cyber-security. Acesso em: out. 2023.

LEZZI, M.; LAZOI, M.; CORALLO, A. Cybersecurity for industry 4.0 in the current literature: A reference framework. **Computers in Industry**, v. 103, p. 97–110, 2018. ISSN 0166-3615. Disponível em: https://www.sciencedirect.com/science/article/pii/S0166361518303658. Acesso em: out. 2023.

OPENPHISH. **OpenPhish**. 2022. Disponível em: https://openphish.com/. Acesso em: out. 2023.

PYTHON SOFTWARE FOUNDATION. **What is Python? Executive Summary**. 2022. Disponível em: https://www.python.org/doc/essays/blurb/. Acesso em: out. 2023.

SILVA, C. M. R. da; FEITOSA, E. L.; GARCIA, V. C. Heuristic-based strategy for phishing prediction: A survey of url-based approach. **Computers & Security**, v. 88, 2020. ISSN 0167-4048. Disponível em: https://www.sciencedirect.com/science/article/pii/S0167404819301622. Acesso em: out. 2023.

SOLIDRUN. **CuBox-i – The little computer that can**. 2017. Disponível em: https://www.solid-run.com/wiki/lib/exe/fetch.php?media=imx6:cubox-i:brochure$_imx6_cubox_2017-09-05.pdf.AcessadoemSetembrode2023$.

SYAFITRI, W. et al. Social engineering attacks prevention: A systematic literature review. **IEEE Access**, v. 10, p. 39325–39343, 2022.

UNIÃO EUROPÉIA. Regulamento nº 679, de 27 de abril de 2016. regulamento (ue) 2016/679 do parlamento europeu e do conselho. **Jornal Oficial da União Europeia**, 2016. Disponível em: https://eur-lex.europa.eu/legal-content/PT/TXT/?uri=CELEX%3A02016R0679-20160504. Acesso em: out. 2023.

APÊNDICE A – SISTEMA DESENVOLVIDO

A.1 LOG_MONITOR.PY

```
from time import time, sleep
from re import search
import tailer
import datetime
import api_conect
from send_email import enviar_email
import csv

caminho_arquivo_log = '/queries.log'
minutos_update = 65 # api atualiza a cada 60 minutos

# Exemplo de uso de notificacao de email:
email_destinatario = 'amontagner28@gmail.com'
email_assunto = 'Notificacao de acesso a URL maliciosa.'
email_corpo = 'Notificacao de acesso a URL maliciosa.'

#################################################################
# Função para lidar com novas linhas adicionadas ao log
def verificar_log(email_destinatario, email_assunto, email_corpo):
    time_inicio = time()
    log_counter = 0
    init = 0
    banco_csv = ''

    nome_arquivo =
    ↪  '/phishing_detector/app/banco_dados_phishing/online-valid_1.csv'
    nome_arq_log_phishing =
    ↪  '/phishing_detector/app/phishing_logs/phishing_queries.log'
    arq_log_count =
    ↪  '/phishing_detector/app/phishing_logs/log_counter.log'

    for linha in tailer.follow(open(caminho_arquivo_log)):
        log_counter +=1
        time_update = time()

```

```
33          if ((init == 1) or ((time_update-time_inicio) >=
            ↪  (60*minutos_update))):
34              #print("Update.....")
35              time_inicio = time_update
36              db_updated = api_conect.update_ph_csv()
37              init = 0
38
39              with open(nome_arquivo, newline='') as arquivo_csv:
40                  banco_csv = csv.reader(arquivo_csv)
41                  url_match = search(r'query:\s\S+', linha)
42                  url_busca = ""
43
44                  if url_match:
45                      url_busca = str(url_match.group(0)[7:])
46                      #url_busca = url_busca[7:]
47                  else:
48                      pass
49                  url = api_conect.busca_url(banco_csv,url_busca)    #
                    ↪  retorna uma url maliciosa
50
51                  if url not in ["",0]:
52                      novo_arquivo_log = open(nome_arq_log_phishing, 'a')
                        ↪  # Cria um novo arquivo de log
53                      novo_arquivo_log.write(f"DNS query:{linha}, Phishing
                        ↪  URL:{url}")
54                      novo_arquivo_log.write('\n')  # Adiciona uma nova
                        ↪  linha após cada entrada
55                      novo_arquivo_log.close()
56                      email_corpo = email_corpo + f" \n Query: {linha}"
57                      email_enviado = enviar_email(email_destinatario,
                        ↪  email_assunto, email_corpo)
58              #print("Updated!!")
59
60          else:
61              #print("Dont update.....")
62              with open(nome_arquivo, newline='') as arquivo_csv:
63                  banco_csv = csv.reader(arquivo_csv)
64                  url_match = search(r'query:\s\S+', linha)
65                  url_busca = ""
```

```
66
67                  if url_match:
68                      url_busca = str(url_match.group(0)[7:])
69                      #url_busca = url_busca[7:]
70
71                  url = api_conect.busca_url(banco_csv,url_busca)    #
                    ↪  retorna uma url maliciosa
72                  if url != 0:
73                      novo_arquivo_log = open(nome_arq_log_phishing, 'a')
                        ↪  # Cria um novo arquivo de log
74                      novo_arquivo_log.write(f"DNS query:{linha}, Phishing
                        ↪  URL:{url}")
75                      novo_arquivo_log.write('\n')  # Adiciona uma nova
                        ↪  linha após cada entrada
76                      novo_arquivo_log.close()
77                      email_corpo = email_corpo + f" \n Query: {linha}"
78                      email_enviado = enviar_email(email_destinatario,
                        ↪  email_assunto, email_corpo)
79
80          save_arq_log = open(arq_log_count, 'a')
81          save_arq_log.write(f"time: {datetime.datetime.now()}, Number of
            ↪  logs analyzed: {log_counter}")
82          save_arq_log.write('\n')
83          save_arq_log.close()
84
85  if __name__ == '__main__':
86      print(f"Monitoring the log file in real time:
        ↪  {caminho_arquivo_log}")
87      verificar_log(email_destinatario, email_assunto, email_corpo)
```

A.2 API_CONECT.PY

```
1  from csv import DictReader
2  from requests import get
3  from os import makedirs, path
4  import subprocess
5  import json
```

```
6
7  def update_ph_csv():
8      url = 'http://data.phishtank.com/data/<api_key>/online-valid.csv'
9      #print("Iniciando atualizacao do banco de dados...")
10     try:
11         comando_cd = f"cd /phishing_detector/app/banco_dados_phishing/
           ↪  && wget {url} && mvl online-valid.csv onine-valid_1.csv"
12         retorno = subprocess.run(comando_cd, check=True, shell=True)
13         if resultado_wget.returncode == 0:
14             #print("Atualizado.")
15             return 0
16         else:
17             raise Exception
18     except:
19         #print('Falha ao atualizar o banco de dados.')
20         return 1
21
22 ## busca de um imput no banco de dados de phishing
23 def busca_url(banco_csv,url_input):
24     for linha in banco_csv:
25         url_maliciosa = linha[1]
26         if url_input in url_maliciosa:
27             #print("\n \n \n")
28             return url_maliciosa
29
30     return 0
31
```

A.3 SEND_EMAIL.PY

```
1  import smtplib
2  from email.mime.multipart import MIMEMultipart
3  from email.mime.text import MIMEText
4
5  # # para habilitar o gmail para mandar email tem que adicionar o 2FA, e
   ↪  habilidat 'senha de app'
6  # # https://support.google.com/accounts/answer/185833?hl=pt-BR
```

```
7
8  def enviar_email(destinatario, assunto, corpo):
9      smtp_host = 'smtp.gmail.com'
10     smtp_port = 587
11     smtp_usuario = 'xxx@xxxx.com'
12     smtp_senha = 'xxxxxxxxxxx'
13
14     # Criar objeto MIME para construir o e-mail
15     msg = MIMEMultipart()
16     msg['From'] = smtp_usuario
17     msg['To'] = destinatario
18     msg['Subject'] = assunto
19
20     msg.attach(MIMEText(corpo, 'plain'))
21     server = smtplib.SMTP(smtp_host, smtp_port)
22     server.starttls()  # Ativar TLS para segurança
23
24     # Autenticar no servidor SMTP
25     server.login(smtp_usuario, smtp_senha)
26
27     # Enviar o e-mail
28     server.sendmail(smtp_usuario, destinatario, msg.as_string())
29
30     # Encerrar a conexão com o servidor SMTP
31     server.quit()
32     return 0
```

A.4 NAMED.CONF

```
1  acl internal {
2      192.168.3.0/24;
3  };
4
5  options {
6      forwarders {
7          1.1.1.1;
8          1.0.0.1;
```

```
    };

    allow-query { internal; };
};



########################
## logging             #
########################

logging {
    channel general {
        file "/general.log" versions 5;
        print-time yes;
        print-category yes;
        print-severity yes;
    };

    channel queries {
        file "/queries.log" versions 5 size 10m;
        print-time yes;
        print-category yes;
        print-severity yes;
    };

    channel security {
        file "/security.log" versions 5;
        print-time yes;
        print-category yes;
        print-severity yes;
    };

    category default { general; };
    category general { general; };
    category config { general; };
    category network { general; };
    category queries { queries; };
    category security { security; };
```

```
};

```

A.5 DOCKERFILE

```
FROM alpine:3.17.5
# Escolhido a imagem alpine:3.17.5 por ser recente e estável até o desenvolvimento deste projeto e
#         por não possui registros de vulnerabilidades no "https://hub.docker.com" até o dia 18/08/2023

########################
# Mantenedor e rotulo #
########################
MAINTAINER Antonio S. Montagner  <antoniomontagner@hotmail.com>
LABEL description='Alpine + Bind9 + Phishing Detector'

######################
## VARIAVEIS ---    #
######################
ENV TZ=America/Sao_Paulo

#####################
## UPDATE & BIND ---#
#####################
# Atualizar a imagem e instalar dependencias
# usando modo debug do shell para acompanhar o processo
# instalar pacote para timezone
RUN set -x \
        && apk update \
        && apk add --no-cache \
                bash \
                bind \
                vim \
                bind-tools \
                tzdata \
                python3 \
```

```
        &&        rm -rf /var/cache/apk/* \
        &&  python -m ensurepip \
        &&  python -m pip install --upgrade pip \
        &&  pip install tailer \
        &&  pip install requests


########################
## Configurar timezone #
########################
RUN ln -snf /usr/share/zoneinfo/${TZ} /etc/localtime && echo ${TZ} >
 ↪  /etc/timezone

########################
## Portas para uso            #
########################
EXPOSE 53/tcp 53/udp

############################
## Copia de arquivos uteis #
############################
COPY ./phishing_detector/ /phishing_detector
COPY ./entrypoint.sh /entrypoint.sh

############################
## Script de inicializacao #
############################
ENTRYPOINT ["/entrypoint.sh"]
```

A.6 DOCKER-COMPOSE.YML

```
version: "3.3"
services:
  my_phishing_project:
    container_name: phishing-detection-and-dns-server
    build:
      context: .
```

```
      dockerfile: ./Dockerfile
    image: bind9_armv7:1.0.1
    ports:
      - "53:53/tcp"
      - "53:53/udp"
    volumes:
      - ./config:/etc/bind
      # Criar volume do projeto
      - ./phishing_detector:/phishing_detector
    restart: unless-stopped
```

A.7 ENTRYPOINT.SH

```
#!/bin/sh

#############################################################################
## MAINTAINER Antonio S. Montagner  <antoniomontagner@hotmail.com>     #
#############################################################################
## Script de inicializacao.                                                 #
#############################################################################
echo "Log de inicializacao, cada linha um restart.." >>
↪  /phishing_detector/named_logs/log_inicializacao.txt

##########################################
## Tornar script da aplicacao executavel #
##########################################
chmod -R +x /phishing_detector

######################
## run as foreground #
######################
python3 /phishing_detector/app/log_monitor.py &

#########################
# iniciar o servico DNS  #
#########################
exec named -f -c /etc/bind/named.conf
```

A.8 TEST_DNS_REQUESTS.PY

```
import subprocess
# Para os testes foi feito este codigo para varrer uma lista de dominios e fazer as requisicoes ao servidor

# os dominios foram pegos do NIC.br e foi inserido aleatoriamente 4 dominios maliciosos
# retirados do banco de dados de phishing.
#
# Quase 150 mil dominios.
#
# dominios em processo de liberacao pelo nic.br
# Mais informacoes em https://registro.br/dominio/processo-de-liberacao/
# Arquivo gerado em 2023-09-11T10:00:09-03:00
arquivo_de_urls = 'urls.txt'

servidor_dns = '192.168.3.16'

# Ler as URLs do arquivo
with open(arquivo_de_urls, 'r') as file:
    urls = file.readlines()

for url in urls:
    url = url.strip()  # Remover espaços em branco e quebras de linha
    comando = f'dig @{servidor_dns} {url}' #consulta DNS

    try:
        resultado = subprocess.check_output(comando, shell=True, text=True)
        #print(f'Resultado da consulta DNS para {url}:\n{resultado}\n')
    except subprocess.CalledProcessError as e:
        #print(f'Erro ao executar o comando para {url}: {e}\n')
print("Terminou.")
```

APÊNDICE B – ARTIGO 1

B.1 UMA BREVE ANÁLISE SOBRE PHISHING

DOI: https://doi.org/10.5902/2448190471731

COMMUNICATIONS AND INNOVATIONS GAZETTE

Uma breve análise sobre phishing

Antonio Silverio Montagner[1], **Carla Merkle Westphal**[1]

[1] Departamento de Informática e Estatística
Universidade Federal de Santa Catarina (UFSC)
88.040-900 – Florianópolis – SC – Brasil

antonio.s.montagner@grad.ufsc.br, carla.merkle.westphall@ufsc.br

Abstract. *The number of phishing attacks reached, in 2021, the highest number of all time until then, being tripled since 2020. This event demonstrates that as phishing techniques are shaped considering the context of their target since its emergence in 1995, these attacks adapt over time. Thus, this problem has persisted in society for more than 25 years and continues to increase its number of victims. With that, this work presents a study about what phishing is, its concepts, its types, and the problems caused in society.*

Resumo. *O número de ataques phishing atingiu em 2021 o maior número de todos os tempos até então, sendo o triplo desde 2020. Tal evento demonstra que, desde seu surgimento em 1995, como as técnicas de phishing são moldadas considerando o contexto de seu alvo, esses ataques acabam se adaptando com o tempo e, dessa forma, tal problema persiste na sociedade há mais de 25 anos e continua aumentando seu número de vítimas. Com isso, este trabalho apresenta um estudo sobre o que é phishing, seus conceitos, seus tipos e os problemas causados na sociedade.*

1. Introdução

Após mais de 25 anos na sociedade, tendo surgido em 1995 [James 2006], o número de ataques de phishing alcançaram mais de 300 mil ataques em dezembro de 2021 [APWG 2021]. Esses ataques exploraram, entre outros cenários, o de pandemia de COVID-19 [Abroshan et al. 2021], período no qual os indivíduos ficaram mais dependentes de serviços online, como aplicativos de mensagens e de mídias sociais, aumentando o número de técnicas de abordagem para se usar no ataque.

Nesse contexto, deve-se considerar que o fator humano por trás do ataque de phishing, que é explorado por formas de engenharia social para aumentar a efetividade do ataque, é um ponto crucial. Dependendo das ações do usuário, pode existir o sucesso ou a falha do ataque [Desolda et al. 2021]. Com isso, a sensação de liberdade, privacidade e segurança acabam sendo comprometidas.

Este artigo terá mais 4 seções: na segunda seção serão apresentados os conceitos básicos sobre phishing, na terceira seção será tratado o que é o phishing e suas características, na quarta seção serão tratados os problemas causados pelo phishing e os temas de pesquisas importantes, e na quinta seção serão descritas as considerações finais sobre o trabalho.

2. Conceitos

Nesta seção serão expostos alguns conceitos importantes relacionados ao ataque de phishing.

2.1. Engenharia Social

Um ataque de engenharia social, segundo [Syafitri et al. 2022], busca manipular uma vítima atacando seu ponto mais fraco. Segundo o dicionário de Inglês de Oxford, o termo ”engenharia social” pode ter dois significados distintos, o primeiro se trata de usar um planejamento centralizado na tentativa de gerenciar uma mudança social e o segundo se trata de persuadir uma pessoa induzindo-a divulgar informações privilegiadas. Deve-se ressaltar que teve a primeira ocorrência de tal termo foi em 1842 e ainda se encontra nos dias atuais [Hatfield 2018]. Alguns tipos de engenharia social, além do phishing, segundo[Syafitri et al. 2022], são: *pretexting*, *baiting* e *ace-to-face interaction*.

Na técnica de *pretexting*, o atacante coleta uma informação pública disponível em *websites*, redes sociais e listas telefônicas, para elaboração de um ataque de comunicação bidirecional, onde o atacante pode oferecer ou pedir algo ao alvo.

A técnica de *baiting* utiliza a curiosidade de membros de uma organização sobre algum item para que conecte tal dispositivo infectado em algum aparelho da organização, assim infectando-o.

A técnica de *ace-to-face interaction* é comumente usada em ataques de engenharia social. Essa técnica explora tirar vantagem de fraquezas psicológicas da vítima, como implorando ajuda ou por acesso a algum dispositivo.

2.2. Fator Humano

Considerando que estamos cercados por tecnologia da qual buscamos o uso para melhorar nossas vidas, também acaba por deixar-nos mais vulneráveis e acessíveis a sistemas enganosos e à exploração. Para considerar um sistema seguro, depende de que as pessoas responsáveis por ele ou com algum nível de relação a ele sigam as normas de segurança impostas. Dessa forma, considerando que pode ocorrer alguns fatores como distração, pressão, estresse e etc, às pessoas, criando fraquezas em um sistema, quais um invasor pode explorar [Desolda et al. 2021].

2.3. Malware

Malware pode ser classificado como um hardware, firmware ou software que, intencionalmente ou não, possam ser inseridos ou adicionados a um sistema e com capacidade de comprometer a confiabilidade, integridade ou disponibilidade dos dados, aplicativos ou do sistema operacional da vítima [Almeida et al. 2022].

2.4. Ransomware

O ransomware é um tipo de malware do qual está diretamente ligado a ataques de phishing e outros ataques de engenharia social, por serem mais direcionados aos usuários finais, com a finalidade de reter dados pessoais e liberá-los apenas mediante ao pagamento de um resgate [Almeida et al. 2022].

3. O que é Phishing

Phishing é um ataque que explora técnicas de engenharia social para realizar um roubo de informações confidenciais [Aleroud and Zhou 2017], estando presente na sociedade desde 1995 [James 2006]. O termo *phishing* se dá ao fato de que o atacante está tentando pescar, do inglês *fishing*, dados; e o "ph" é derivado de sofisticado, do inglês *sophisticated*, por conta das técnicas mais sofisticadas que tais atacantes usam para se distinguir da atividade mais simples de pescar [James 2006].

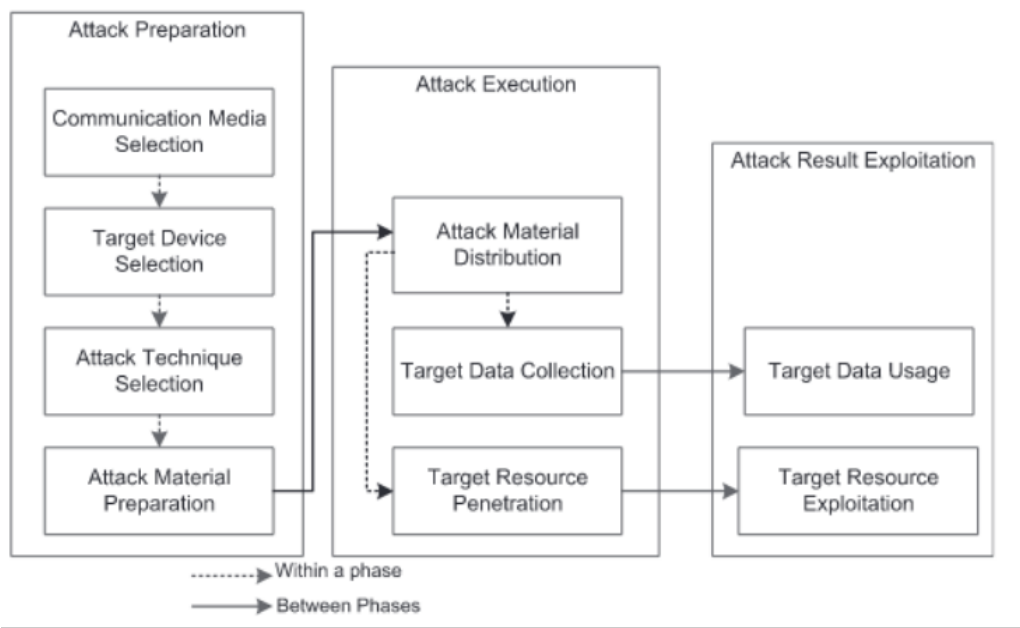

Figura 1. Fases do processo de um ataque *phishing*. [Aleroud and Zhou 2017]

3.1. Processo de elaboração

Passando por um processo, assim como da Figura 1, onde se tem as fases de preparação, execução e exploração dos resultados. Cada fase tem seus passos a serem seguidos.

Na primeira, tem que selecionar a mídia a ser usada, que seria o meio a ser usado, a seleção do dispositivo alvo, que é corresponde a seleção do vetor, a técnica de abordagem que vai ser aplicada e, por fim, a preparação do material para o ataque. Após isso, tem a execução do ataque, onde se distribui o material contaminado colocando na coleção de dados do alvo ou em seus recursos. Com isso, caso o alvo usar tal dado ou explorar tal recurso e acabar fornecendo dados sensíveis ou relevantes ao atacante. Caso o ataque seja realizado em conjunto outros tipos de ataques, como o de ransomware, assim, contaminando o dispositivo.

Na Figura 2, pode-se ver alguns exemplos de meios, vetores e a técnica de abordagem para a elaboração de um ataque de *phishing*.

3.1.1. Meios

Para que um ataque de *phishing* aconteça, é necessário ter um meio para haver a interação entre o atacante e o alvo, e, para que isso aconteça, existem alguns meio mais comuns, que seria o meio da internet, *Short Messaging Service* (SMS), meios dos quais pessoas utilizam normalmente e que podem ser usados por atacantes para interagir com vitimas [Chiew et al. 2018].

3.1.2. Vetores

Existem vários vetores associados aos meios citados anteriormente dos quais servem para intermediar um ataque por um meio e uma abordagem escolhida, contudo, vetores associados ao meio de internet são os mais populares entre os ataques de *phishing* [Chiew et al. 2018]. Pode-se destacar os vetores de Email, para internet, Smishing, para SMS, e Vishing, para serviços de voz.

3.1.3. Abordagens Técnicas

Existem várias abordagens técnicas que podem ser usadas em um ou mais vetores para a implementação de um ataque de *phishing*, assim como mostrado na Figura 2. Na sequência do texto são explicados cada um dos tipos de abordagens com base na referência [Chiew et al. 2018].

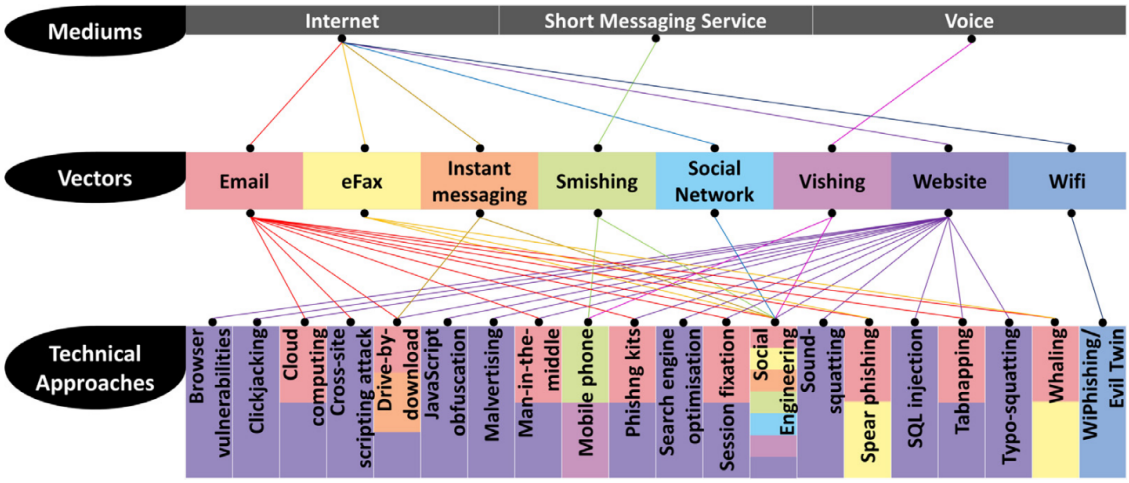

Figura 2. A interligação entre meio, vetor e abordagem das técnicas de *phishing*. [Chiew et al. 2018]

Browser vulnerabilities Trata-se de explorar vulnerabilidades de um browser para lançar um ataque de *phishing* a um usuário, por exemplo por meio de extensões e *plug-ins* de fornecedores externos.

Clickjacking Também conhecido como *user interface (UI) redressing attack*, trata-se da manipulação da UI de uma página web, onde possibilita uma ação externa ao utilizar a página.

Cloud computing Trata-se de um serviço online que está aumentando a popularidade e que podem ser atacados de 6 formas, atacando os as relações entre os três componentes que compõem tal serviço, que são os usuários, o serviço e o provedor da nuvem. Relações mostradas na Figura 3.

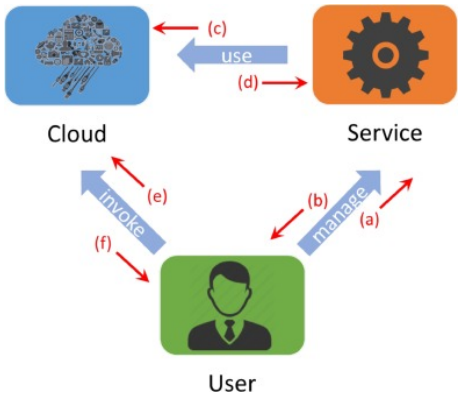

Figura 3. A relação entre os três componentes da computação em nuvem. [Chiew et al. 2018]

Cross-site scripting (XSS) attack Trata-se da exploração de uma vulnerabilidade de *websites* onde aceitam que o atacante injete um código malicioso dentro de algum campo de dados do site e ele interprete tal código podendo permitir que acesse informações pessoais como login e credenciais.

Drive-by-download Trata-se de injetar um malware, vírus ou código em uma máquina apenas visitando um *websites*, vendo um HTML ou recebendo um email.

javascript obfuscation Trata-se de utilizar JavaScript para mascarar a barra de endereços, barra de *status*, barra de ferramentas ou a área de *menu*, assim conseguindo falsificar os endereços ou *status* de tais áreas.

Malvertising Trata-se de utilizar o serviço de hospedagem online de anúncios como meio de distribuir malware, onde, ao clicar no anúncio, um malware dinâmico infecta a máquina da vítima e explorar suas vulnerabilidades com o objetivo de roubar informações pessoais.

Man-in-the-middle (MITM) Trata-se de o atacante se colocar no meio da comunicação entre a vítima e uma aplicação *web*, assim como na Figura 4, onde o atacante é capaz de controlar as informações submetidas da vítima à aplicação *web*, possibilitando a captura

de credenciais de autenticação. Tal captura pode ser feita por meio da utilização de um proxy transparente, onde o atacante tem acesso a essa rede e ao tráfego dela, demonstrado na Figura 5.

Figura 4. Exemplo de um ataque Man-in-the-Middle. [Chiew et al. 2018]

Figura 5. Exemplo de utilização de um proxy transparente em um ataque Man-in-the-Middle. [Chiew et al. 2018]

Mobile phone Trata-se de distribuir aplicativos maliciosos para aparelhos de telefones móveis, onde buscam controlar a transferência de dados entre as aplicações do dispositivo, possibilitando a captura de informações pessoais.

Phishing kits Trata-se de ferramentas que possibilitam que o atacante crie um *websites*, emails e *scripts* para obter dados inseridos por um usuário sem ser necessário conhecimento de programação para isso.

Search engine optimisation Trata-se de otimizar a entrega *websites* de *phishing* para potenciais vítimas usando técnicas de otimização para ferramentas de busca.

Session fixation Trata-se de roubar identificadores de sessões, como cookies, gerados quando um usuário faz login em um site com esse tipo de falha de segurança, assim, o atacante é capaz de usar a seção do usuário para efetuar atividades maliciosas como transferência de dinheiro.

Social engineering Trata-se da utilização de técnicas de engenharia social (seção 2.1) para obter vantagem sobre a vítima.

Sound-squatting Trata-se de registrar domínio de sites com nomes similares aos de sites legítimos, assim, o atacante tira vantagem da confusão do usuário que é redirecionado a uma versão de *phishing* do *website* do qual pretendia acessar.

Spear phishing Trata-se de um ataque direcionado a um indivíduo, um grupo ou organização, onde se desenvolve um email com conteúdo relevante e que a vítima conhece o remetente, assim, evitando alguma suspeita da vítima e possibilitando efetuar solicitações de detalhes de login ou rodar algum conteúdo com malware.

SQL injection Trata-se da exploração de uma vulnerabilidade do banco de dados onde é possível injetar comandos e capturar dados da tabela de dados do usuário.

Tabnapping Trata-se do roubo de abas do navegador, onde o atacante envia um link em um email, do qual é aberto no navegador da vítima e, tal site, possui um código JavaScript do qual monitora a atividade do navegador da vítima e também carrega uma tela de login conhecida pela vítima, como do Gmail, para ela achar q a seção foi fechada e que é necessário conectar-se novamente.

Typo-squatting Trata-se de ataque que o atacante registra nomes de domínio com possíveis erros de digitação que o usuário possa fazer, tal como ”www.mybank.br” e ”wwwmybank.br”, assim possibilitando o acesso acidental ao site malicioso e descarregando um malware no dispositivo da vítima.

Whaling Trata-se de um tipo de *spear phishing* do qual tem como alvo pessoas de alto nível executivo e com altos privilégios de acesso na organização, onde, por meio de um malware, o atacante tem acesso a uma porta de acesso ao sistema da organização.

Wiphishing / Evil Twin Trata-se de um ataque que usa redes sem fio, onde o atacante se coloca entre o usuário e a verdadeira rede sem fio, possibilitando que o atacante seja capaz de espionar os dados enviados e recebidos pelo usuário.

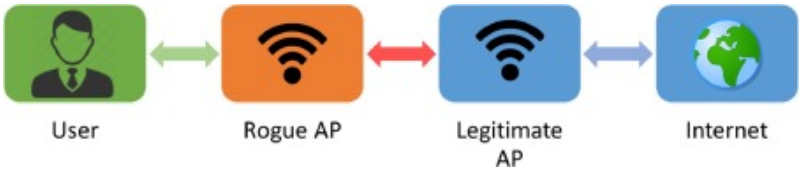

Figura 6. Ataque WiPhishing/Evil Twins. [Chiew et al. 2018]

4. Phishing durante os anos de 2020 e 2021

Problemas com *phishing* persistem na sociedade há mais de 25 anos [James 2006]. Além da persistência, o número desse tipo de fraude vem crescendo nos últimos anos, atingindo o maior número de ataques de todos os tempos em Dezembro de 2021 [APWG 2021]. Com isso, nesta seção serão expostos alguns problemas, sobre *phishing*, encontrados na literatura entre os anos de 2020 e 2021.

4.1. Acontecimentos Práticos

Com a pandemia de COVID-19 em 2020, despertou um novo nível de ansiedade, medo e *stress* na sociedade, e, com isso, pessoas mal intencionadas buscam tirar proveito de pessoas que possuem tais problemas, facilitando os ataques de *phishing*.

Nesse contexto de pandemia, também houve o aumento no número de indivíduos dependentes de serviços online, como compras, videochamadas entre outros, com isso, deixando as pessoas mais vulneráveis a fraudes online, tal como envio de malwares, dos quais 94% é enviado por e mail e, desses, 32% envolve *phishing*.

Com tudo isso, invasores cibernéticos aproveitaram para tirar vantagens de possíveis vítimas ao usar informações e palavras chaves relacionadas ao COVID-19, para realizar ataques de *phishing* [Abroshan et al. 2021].

A Figura 7 demonstra algumas variáveis que influenciaram na probabilidade de acontecer um ataque de phishing durante a pandemia de COVID-19. Nota-se que, com mais vetores hipotéticos (siglas H1, H2 e H3 da figura), maior a possibilidade de acontecer um *phishing*, já que aumenta o número de abordagens, como tratado na seção 3.1.

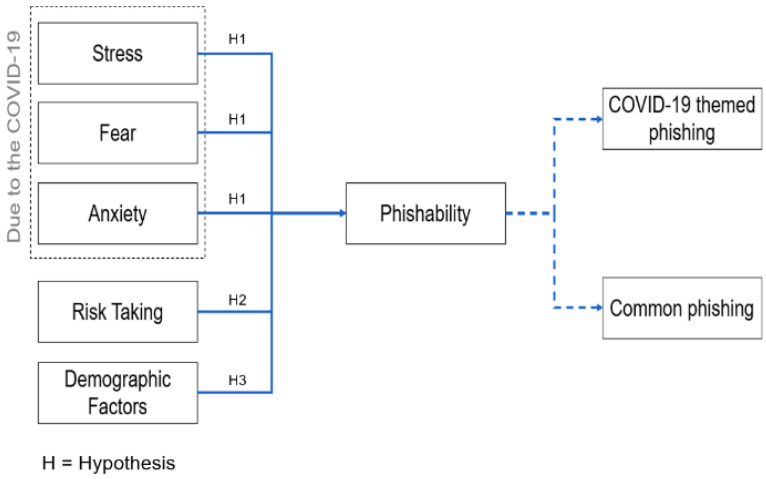

Figura 7. Grafo do processo de efetivação de um ataque *phishing*. [Abroshan et al. 2021]

Com tudo isso, e considerando o ano de 2021 e também a continuação da pandemia de COVID-19, dados demonstram o crescente número de ataques de *phishing* no decorrer do ano (Figura 8), até atingir o maior valor histórico até então, em dezembro, assim corroborando o efeito dos problemas supracitados.

No Brasil, durante esse período, houve uma leve queda no número de ataques, como pode-se ver na Figura 9. No entanto, os números ainda são preocupantes por conta de serem em um período de 3 meses. Dentre esses números, a quantidade de *phishing* contra empresas de SaaS e Webmail totalizam 18% dos ataques no terceiro trimestre e contra empresas de comércio eletrônico foram 37,5% [APWG 2021].

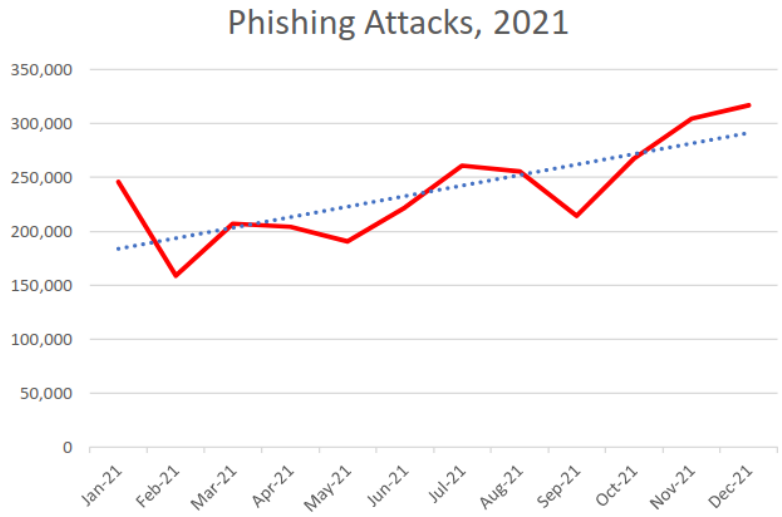

Figura 8. Números de ataques *phishing* em 2021. [APWG 2021]

Figura 9. Números de ataques *phishing* no Brasil em 2021. [APWG 2021]

4.2. Trabalhos relacionados

Sobre pesquisas na área de detecção de *phishing*, destacam-se algumas técnicas, tais como heurísticas e *Machine Learning*.

Um exemplo de técnica heurística foi desenvolvida em [Almeida et al. 2022], trabalho desenvolvido com base em *web crawling*, que é um processo usado pelos motores de busca para coletar páginas da *web* [Almeida et al. 2022]. Neste trabalho, também é possível monitorar o comportamento do usuário, detectando suas ações e determinando o momento em que o usuário acessou o conteúdo malicioso.

Um exemplo de técnica de machine learning, é descrita em [Sahingoz et al. 2019], que considera o uso de diferentes algoritmos de aprendizado de máquina, como *Decision Tree* que se trata de um algoritmo que usa um modelo de decisões em forma de árvore e suas possíveis consequências[1]. O trabalho implementa uma forma de detecção em tempo real de páginas *web* de *phishing* investigando sua URL. Tal trabalho, tem como foco não apenas executar os algoritmos de aprendizado de máquina, mas também a extração de características importantes de tais dados para compor seu próprio conjunto de dados.

Outras técnicas de inteligência artificial, como *deep learning* e *Hybrid learning* são apresentadas em [Basit 2021], para implementação de formas de detecção de *phishing*.

5. Considerações finais

Neste trabalho são discutidas as principais características de um ataque de *phishing*. Também são mostrados dados sobre o problema causado na sociedade por tal tipo de ataque. Com isso, considera-se que *phishing* é um problema persistente na sociedade há anos e com muitas possibilidades de evolução quanto ao combate e mitigação.

Foi explicitada a estrutura do *phishing*, como ela é preparada, considerando todo um processo de seleção de meio, vetor e técnica de abordagem, seguido da execução do ataque mediante a técnica escolhida e, por fim, a exploração dos resultados obtidos. Em relação às pesquisas relacionadas, nota-se que existem pesquisas bem atuais, utilizando as mais recentes formas computacionais e que sempre há espaço para aperfeiçoamentos.

Por fim, não deve-se desconsiderar o *phishing* e deixar de pesquisar meios de detecção e mitigação pois tal ataque persiste há mais de 25 anos e deverá persistir por ainda mais tempo, já que existem sempre novas maneiras e técnicas de ataques a serem combatidas. Dessa forma, este trabalho surge como uma fonte de fácil compreensão para introduzir o tema proposto, proporcionando a abertura de um leque de oportunidades para a aprofundamento no conteúdo.

[1] https://en.wikipedia.org/wiki/Decision_tree

Referências

Abroshan, H., Devos, J., Poels, G., and Laermans, E. (2021). Covid-19 and phishing: Effects of human emotions, behavior, and demographics on the success of phishing attempts during the pandemic. *IEEE Access*, 9:121916–121929.

Aleroud, A. and Zhou, L. (2017). Phishing environments, techniques, and countermeasures: A survey. *Computers Security*, 68:160–196.

Almeida, R. A. O. C. B. d. et al. (2022). Heuristic phishing detection based on web crawling and user behaviour monitoring with a deterministic approach for cybersecurity.

APWG (2021). Phishing activity trends report 4th quarter 2021. In *PHISHING ACTIVITY TRENDS REPORT 4th Quarter 2021*. apwg.org.

Basit, A. e. a. (2021). A comprehensive survey of ai-enabled phishing attacks detection techniques. *Telecommunication Systems*.

Chiew, K. L., Yong, K. S. C., and Tan, C. L. (2018). A survey of phishing attacks: Their types, vectors and technical approaches. *Expert Systems with Applications*, 106:1–20.

Desolda, G., Ferro, L. S., Marrella, A., Catarci, T., and Costabile, M. F. (2021). Human factors in phishing attacks: A systematic literature review. 54(8).

Hatfield, J. M. (2018). Social engineering in cybersecurity: The evolution of a concept. *Computers Security*, 73:102–113.

James, L. (2006). Chapter 1 - banking on phishing. In James, L., editor, *Phishing Exposed*, pages 1–35. Syngress, Burlington.

Sahingoz, O. K., Buber, E., Demir, O., and Diri, B. (2019). Machine learning based phishing detection from urls. *Expert Systems with Applications*, 117:345–357.

Syafitri, W., Shukur, Z., Mokhtar, U. A., Sulaiman, R., and Ibrahim, M. A. (2022). Social engineering attacks prevention: A systematic literature review. *IEEE Access*, 10:39325–39343.

APÊNDICE C – ARTIGO 2

C.1 SISTEMA DE DETECÇÃO DE PHISHING COM BASE EM QUERY DNS

DOI: https://doi.org/10.5902/2448190485266

Sistema de detecção de phishing com base em query DNS

**Antonio Silverio Montagner, Carla Merkle Westphall,
Rômulo A. O. C. B. de Almeida, Guilherme Eliseu Rhoden**

[1] Departamento de Informática e Estatística
Universidade Federal de Santa Catarina (UFSC)
88.040-900 – Florianópolis – SC – Brasil

antonio.s.montagner@grad.ufsc.br, carla.merkle.westphall@ufsc.br,

romulob00@hotmail.com, guilherme.rhoden@rnp.br

Abstract. *Phishing, a social engineering technique by which attackers exploit victims' weaknesses to obtain confidential information, is a problem that has plagued society for more than two decades. With the constant evolution of communication technologies and the increase in interactivity between people, cyber attack tactics, including phishing, have also improved. This sophistication has led to an increase in the effectiveness of this type of attack, making it one of the most prominent threats faced by Internet users.*
Given this persistent challenge, researchers have strived to create efficient solutions for detecting and mitigating this type of attack. In this context, the present work proposes the development and demonstration of a phishing detection system developed under an ARM architecture, offering an easy-to-deploy implementation using Docker technology. Furthermore, the system makes use of a DNS server log service as an integral part of its detection strategy. This initiative aims to contribute to cyber protection, offering a robust and practical approach to identifying and preventing phishing attempts, seeking to reduce this growing problem.

Resumo. *O phishing, uma técnica de engenharia social pela qual os atacantes exploram os pontos fracos das vítimas para obter informações confidenciais, é um problema que assola a sociedade há mais de duas décadas. Com a constante evolução das tecnologias de comunicação e o aumento da interatividade entre as pessoas, as táticas de ataque cibernético, incluindo o phishing, também se aprimoraram. Essa sofisticação levou a um aumento na eficácia desse tipo de ataque, tornando-o uma das ameaças mais proeminentes enfrentadas pelos usuários da Internet.*
Em vista desse persistente desafio, pesquisadores têm se empenhado na criação de soluções eficientes para a detecção e mitigação desse tipo de ataque. Nesse contexto, o presente trabalho propõe o desenvolvimento e demonstração de um sistema de detecção de phishing desenvolvido sob uma arquitetura ARM, oferecendo uma implementação de fácil implantação por intermédio da tecnologia Docker. Além disso, o sistema faz uso do serviço de logs de um servidor DNS como parte integrante de sua estratégia de detecção. Essa iniciativa visa contribuir para a proteção cibernética, oferecendo uma abordagem robusta e prática para a identificação e prevenção de tentativas de phishing, buscando diminuir esse problema crescente.

1. Introdução

Como apresentado em [Montagner and Westphall 2022], o fenômeno do *phishing*, um tipo de ataque cibernético que explora aspectos sociais e psicológicos das vítimas, tem observado um incremento ao longo dos anos, especialmente após o período pandêmico em 2021.

O *phishing* é notoriamente danoso, uma vez que pode induzir suas vítimas a tomar decisões precipitadas, como o download de arquivos maliciosos ou a divulgação de informações pessoais. Como contramedida a esse problema em ascensão, estudos como os apresentados em [de Almeida 2022] e [da Silva et al. 2020] têm se dedicado a desenvolver novos métodos ou aprimorar os já existentes de detecção de *phishing*.

Nesse contexto, o presente trabalho propõe o desenvolvimento e demonstração de um sistema de detecção de *phishing* desenvolvido sob uma arquitetura ARM . Tal proposta contempla a implementação simplificada por meio da tecnologia Docker. Além disso, o sistema utiliza do serviço de registro de logs de um servidor DNS e de um banco de dados de URLs maliciosas, complementado por uma aplicação desenvolvida em Python, para estabelecer um robusto sistema de detecção.

Este artigo está estruturado em quatro seções adicionais. Na segunda seção, serão discutidos os conceitos fundamentais relacionados ao *phishing*. A terceira seção abordará o desenvolvimento detalhado do sistema em questão. A quarta seção se encarregará da análise e discussão dos resultados obtidos. Por fim, a quinta seção conterá as considerações finais a respeito deste estudo.

2. Conceitos

Nesta seção serão expostos alguns conceitos importantes relacionados ao trabalho.

2.1. Phishing

Phishing é um ataque que explora técnicas de engenharia social para realizar um roubo de informações confidenciais [Aleroud and Zhou 2017], estando presente na sociedade desde 1995 [James 2006]. O termo *phishing* se dá pelo fato de que o atacante está tentando pescar, do inglês *fishing*, dados; e o 'ph' é uma derivação de *sophisticated*, palavra em inglês cuja a tradução é 'sofisticado', por conta das técnicas mais sofisticadas que tais atacantes usam para se distinguir da atividade mais simples de pescar [James 2006].

Meios Para que um ataque de *phishing* aconteça, é imprescindível estabelecer um canal de comunicação entre o atacante e o alvo. Nesse contexto, diversos meios tradicionais se destacam, sendo a Internet e o *Short Messaging Service* (SMS) os mais notáveis exemplos. Essas são vias frequentemente utilizadas pelas pessoas em seu cotidiano e, lamentavelmente, podem ser exploradas por atacantes como meio de interação com suas potenciais vítimas, conforme abordado em [Chiew et al. 2018].

Vetores Diversos vetores estão correlacionados aos meios mencionados anteriormente, desempenhando o papel de intermediários entre um meio e uma abordagem selecionada. No entanto, nota-se que os vetores vinculados ao meio da Internet se destacam como os mais prevalentes nos ataques de *phishing* segundo [Chiew et al. 2018].

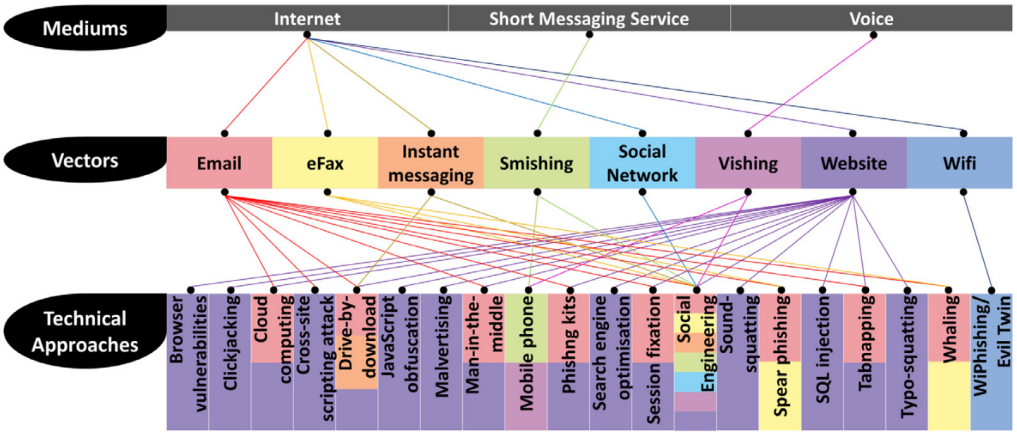

Figura 1: A interligação entre meio, vetor e abordagem das técnicas de *phishing*. [Chiew et al. 2018]

Abordagens Técnicas Diversas abordagens técnicas podem ser empregadas em um ou mais vetores como parte da concepção de um ataque de *phishing*, conforme ilustrado na Figura 1. O resultado dessa ação pode conduzir a eventual exposição de dados sensíveis ou relevantes por parte do alvo, caso esse venha a fazer uso dessas informações ou explorar recursos específicos.

Importante observar que, em certos cenários, o ataque de *phishing* pode ser orquestrado em concomitância com outros tipos de investidas, como o ataque de *ransomware*, resultando na potencial contaminação do dispositivo alvo.

2.2. Bancos de dados de Phishing

Os banco de dados de *phishing* se tratam de acervos de dados e informações coletadas na Internet sobre *phishing*. Neste trabalho será usado o banco de dados *PhishTank* tal como o *Phishload* e *Openphish*.

O *PhishTank* é um banco de dados colaborativo de *phishing* disponível na Internet. Esse sistema fornece uma API[1] aberta para desenvolvedores. Seus dados são fornecidos em vários formatos e atualizados de hora em hora, assim colaborando com uma aplicação mais rápida e atualizada na detecção de *phishing* [Systems 2022].

2.3. DNS & BIND9

Segundo a RFC 1034 [Network Working Group 1987], o *Domain Name System* (DNS) é um sistema fundamental da Internet projetado para resolver o desafio de associar nomes de domínios legíveis por humanos, como ”www.exemplo.com”, a endereços IP numéricos, como ”192.0.2.1”. Esse sistema de nomenclatura hierárquico e distribuído é essencial para a funcionalidade da Internet, criando uma estrutura de árvore que organiza domínios e subdomínios. Dessa forma, facilitando a resolução de nomes, permitindo que servidores consultem uns aos outros para encontrar o IP correto para o domínio que está procurando.

O BIND9, servidor de nomes de domínios (DNS) de código aberto, é amplamente reconhecido por sua versatilidade, permitindo que um único servidor DNS desempenhe

[1]APIs são mecanismos que permitem que dois componentes de software se comuniquem usando um conjunto de definições e protocolos [**?**].

diversas funções, incluindo a de servidor de nomes autoritativo, resolvedor e, em sistemas suportados, até mesmo como resolvedor de encaminhamento. Sua reputação sólida é sustentada por sua confiabilidade, robustez e flexibilidade, tornando-o apto a atender às exigências de uma variada gama de implementações de DNS, desde redes locais de pequeno porte até ambientes globais de alto tráfego [Internet Systems Consortium 2023].

2.4. Docker & Docker Compose

O *Docker*, um software de código aberto, se destaca como uma plataforma que capacita os desenvolvedores a automatizar o ciclo de vida das aplicações, abrangendo desde sua criação até sua execução em ambientes isolados. Nesse contexto, são fornecidos ambientes virtuais que englobam todos os elementos necessários para garantir a execução consistente e independente das aplicações, desde o estágio de desenvolvimento até a produção. O *Docker* oferece uma solução eficiente para isolar aplicativos e suas dependências, viabilizando a movimentação e replicação descomplicada entre diferentes contextos [Docker 2023].

Por sua vez, o *Docker Compose* é uma ferramenta que simplifica a definição e gestão de aplicações que operam no ambiente *Docker*. Ele possibilita descrever de maneira eficaz a infraestrutura de uma aplicação e suas interações entre os componentes, tornando mais acessíveis os processos de implantação e escalabilidade de aplicações mais complexas. Além disso, o *Docker Compose* também permite a configuração de ambientes de desenvolvimento locais que se assemelham aos ambientes de produção, garantindo uniformidade ao longo do ciclo de vida do desenvolvimento de software [Docker 2023].

3. Proposta e desenvolvimento de um sistema de detecção de phishing com base em query DNS

Nesta seção serão expostos a arquitetura e as características do desenvolvimento usados para a elaboração de um *software* para detecção de *phishing*.

3.1. Arquitetura do Sistema

Na Figura **??**, é apresentada a topologia do sistema proposto. O fluxo de execução da análise de um domínio tem início com a solicitação de resolução DNS por parte de um usuário, representado como *Residential Private Networks*, direcionada ao servidor Bind9 em execução no sistema de virtualização em contêineres, conforme destacado no ponto (1). Após receber essa requisição, o servidor irá fornecer o endereço IP associado ao domínio solicitado, conforme referenciado no ponto (2), ao mesmo tempo em que registra os devidos dados dessa operação, conforme descrito no ponto (3).

Quando um novo registro é inserido no arquivo de log, a aplicação detecta e analisa o texto com o objetivo de extrair o domínio solicitado, conforme referenciado no ponto (4). Em seguida, a aplicação verifica se esse domínio é malicioso ou não, baseando-se em um banco de dados de URLs maliciosas obtido por meio de uma API do *PhishTank*, conforme indicado no ponto (5). Se o domínio não for considerado malicioso, nenhum processo adicional é executado. Entretanto, se for identificado como um domínio malicioso, os dados relativos à solicitação maliciosa são registrados em um novo arquivo de log, como indicado no ponto (6). Além disso, um incidente desse tipo é comunicado por meio de uma notificação por e-mail, conforme descrito no ponto (7).

A seguir, serão explicitados os componentes principais desse sistema.

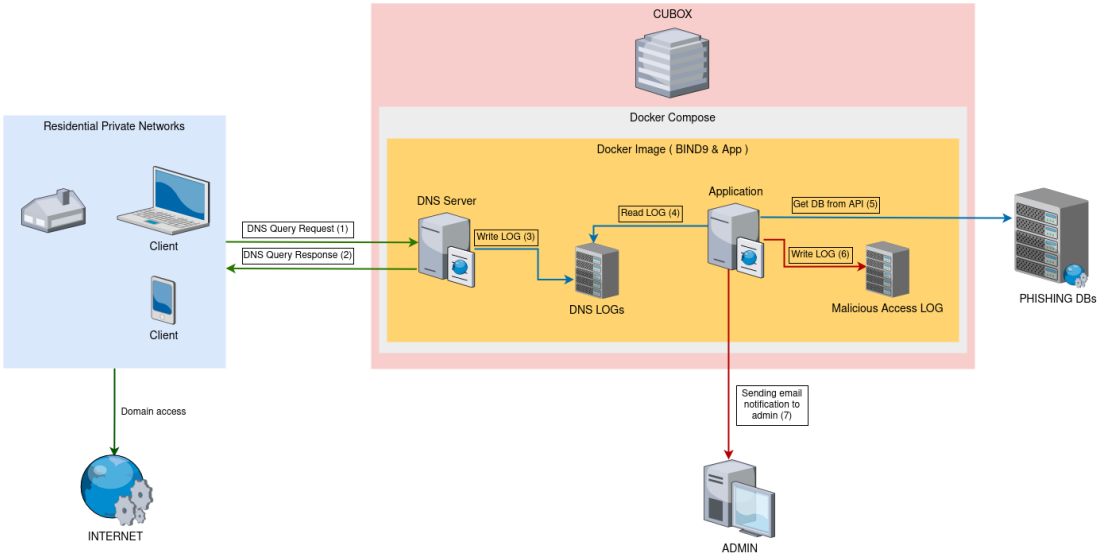

Figura 2: Topologia do sistema desenvolvido.

3.1.1. Componentes Principais

CUBOX Para todos os testes, utilizou-se um minicomputador CuBox, caracterizado por ser uma máquina de hardware robusto e de eficiência energética, capaz de oferecer suporte a uma ampla variedade de aplicações [SolidRun 2017]. Os dados a seguir são as características do CUBOX utilizado.

```
Modelo:
    sr-imx6
CPU:
    Arquitetura: armv7l
    Modelo Cortex-A9
    4 CPUs / 996 MHz
Memoria RAM:
    2011MiB
Interface Ethernet:
    nome logico: eth0
    capacidade: 1 Gbit/s
Armazenamento:
    7.4G
Sistema Operacional:
    Linux sr-imx6 5.10.0-16-armmp
```

Figura 3: Ficha técnica CuBox.

DNS Server Para o sistema proposto, foi escolhido o uso de um servidor DNS, mais especificamente o uso do BIND9, por ser um serviço leve, compatível com a imagem Alpine para Docker, de fácil implementação, de fácil uso por parte do cliente final por não precisar de uma instalação de um agente de monitoramento e apenas necessitar da alteração do servidor DNS da máquina que quer analisar e por possuir o recurso de gerar registros de logs das requisições.

Application Trata-se de uma aplicação Python [Python Software Foundation 2022], linguagem de programação escolhida por comodidade e facilidade na implementação, en-

carregada de baixar a base de dados do *PhishingTank*, que será utilizada para a verificação de domínios maliciosos. Essa aplicação também é responsável por monitorar os registros de log de domínios gerados pelo servidor DNS, a fim de rastrear os domínios acessados pelos usuários. Dessa forma, ao atualizar a base de dados e analisar os registros de log DNS, a aplicação busca o domínio registrado nos logs de DNS na base de dados. Se o domínio for encontrado, ele é registrado em um novo arquivo de log de acesso malicioso (*Malicious Access LOG*) contendo a consulta DNS realizada e o domínio malicioso acessado. Posteriormente, um e-mail de notificação é enviado para um usuário predefinido.

Script de inicialização Um *Shell script* foi desenvolvido com o propósito de configurar e iniciar a aplicação Python, juntamente com o servidor BIND, fornecendo previamente todos os parâmetros necessários para garantir sua inicialização adequada.

Docker e Docker Compose Os serviços *Docker* e *Docker Compose* desempenham um papel fundamental na configuração e inicialização adequada de todos esses serviços mencionados anteriormente.

O Dockerfile apresenta as seguintes características de configuração do sistema:

- Utilização de uma imagem Alpine como base, devido à compatibilidade com os softwares que serão utilizados, sua leveza e capacidade de executar em arquitetura ARMv7.
- Instalação de todas as dependências básicas, incluindo a atualização do fuso horário, Python, entre outras, além da instalação do serviço BIND.
- Estabelecimento de um volume contendo os arquivos necessários para o projeto, a serem copiados para o sistema.

Para facilitar a inicialização e montagem desse sistema, a configuração é realizada por meio de um arquivo de configuração do *Docker Compose*, que possui as seguintes características:

- Definição de volumes para a inicialização.
- Especificação da execução de um *Shell script* responsável por iniciar as aplicações.

3.1.2. Coleta de Dados

O sistema Python está constantemente monitorando as entradas do arquivo de log do servidor DNS. Ao encontrar uma entrada na Figura 4 específico, ele realiza uma análise minuciosa, recortando o conteúdo após ”query:”. O resultado desse recorte, denominado ”url”, é então encaminhado para o método de consulta no banco de dados. Esse já tem previamente carregado com um conjunto de dados relacionados a *phishing* e, neste momento, realiza uma pesquisa para verificar a existência desse domínio. Se o domínio for identificado como pertencente a uma URL maliciosa, por exemplo, ”https://dominio.com”, que pode ser parte de ”https://dominio.com/sub_caminho_malicioso”, o domínio juntamente com o registro de log DNS é arquivado em um novo registro referente a domínios maliciosos, seguindo a Figura 5. Após essa etapa, é gerado e enviado um e-mail na Figura 6 para uma conta registrada, a fim de informar sobre o incidente de segurança que ocorreu.

```
------------------------------------------------------------
DNS query:01-Sep-2000 21:47:42.071 queries: info: client
@0xb5a122d5 192.168.3.9#60893 (https://malicious.com.br):
query: https://malicious.com.br IN A +E(0)K (172.23.0.2)
------------------------------------------------------------
```

Figura 4: Exemplo de log DNS.

```
------------------------------------------------------------
DNS query:01-Sep-2000 21:47:42.071 queries: info: client
@0xb5a122d5 192.168.3.9#60893 (https://malicious.com.br):
query: https://malicious.com.br IN A +E(0)K (172.23.0.2),
Phishing URL: https://malicious.com.br/olde/
------------------------------------------------------------
```

Figura 5: Exemplo de log malicioso.

```
------------------------------------------------------------
Subject: Notificacao de acesso a URL maliciosa.

from: sender@email.com
to: receiver@email.com

Content:
Notificacao de acesso a URL maliciosa.
Query: 01-Sep-2000 21:47:42.071 queries: info: client
@0xb5a122d5 192.168.3.9#60893 (https://malicious.com.br):
query: https://malicious.com.br IN A +E(0)K (172.23.0.2)
------------------------------------------------------------
```

Figura 6: Exemplo de notificação de e-mail.

3.2. Desenvolvimento do Sistema de Análise de Phishing

Para a implementação do sistema proposto, foi necessário seguir os passos mostrados abaixo.

named.conf A Figura 7 se trata da configuração usada no servidor Bind9. Foram definidos os servidores DNS que o nosso servidor local irá buscar quando não for capaz de resolver o domínio pedido, definido os parâmetros para que o servidor gere logs de todas as resoluções que ele responder e também foi adicionado uma regra para que o servidor apenas resolva requisições de IPs dentro de uma rede especificada.

```
------------------------------------------------------------
acl internal {
    192.168.3.0/24;
};
options {
    forwarders {
        1.1.1.1;
        1.0.0.1;
    };
    allow-query { internal; };
};
logging {
    channel general {
        file "/general.log" versions 5;
        print-time yes;
        print-category yes;
        print-severity yes;
    };
    category security { security; };
};
------------------------------------------------------------
```

Figura 7: named.conf

Estrutura da aplicação Python A aplicação é composta por três arquivos Python principais, um arquivo CSV contendo os domínios maliciosos e dois arquivos de registro, assim como mostra a Figura 8. O arquivo *log_monitor.py* desempenha um papel central, responsável por carregar o banco de dados armazenado em *onlinevalid_1.csv*. Ele monitora e analisa os registros gerados pelo servidor DNS em busca de domínios solicitados, encaminhando qualquer domínio encontrado para o método de verificação em *api_conect.py*. Além disso, mantém um registro do número de solicitações processadas pela aplicação em *log_counter.log*. No caso de um domínio ser identificado como malicioso, os dados são registrados em *phishing_queries.log*, e um método em *send_email.py* é invocado para enviar uma notificação por email.

```
------------------------------------------------------------
-- app
   |-- banco_dados_phishing
   |   `-- online-valid_1.csv
   |-- phishing_logs
   |   |-- log_counter.log
   |   `-- phishing_queries.log
   |-- api_conect.py
   |-- log_monitor.py
   `-- send_email.py
------------------------------------------------------------
```

Figura 8: Topologia de arquivos da aplicação.

entrypoint.sh Para configurar e iniciar as aplicações, foi necessário estruturar o seguinte *Shell script*, assim como mostrado na Figura 9.

```
------------------------------------------------------------
#!/bin/sh
# Tornar script da aplicacao executavel
chmod -R +x /phishing_detector
# rodar em background
python3 /phishing_detector/app/log_monitor.py &
# iniciar o servico DNS
exec named -f -c /etc/bind/named.conf
------------------------------------------------------------
```

Figura 9: entrypoint.sh

Docker e Docker Compose Tendo os softwares do *Dockerfile* e *Docker Compose* instalados segundo sua documentação, o *Dockerfile* foi configurado da seguinte forma: foi escolhida a imagem Alpine devido à sua compatibilidade com o projeto; em seguida, foram definidas as dependências necessárias para o projeto, incluindo o Python para a execução da aplicação de verificação de logs e a instalação das bibliotecas necessárias; o servidor DNS BIND foi instalado para servir como servidor DNS, o tzdata foi configurado para ajustar o fuso horário do sistema; a porta necessária para o servidor DNS foi configurada para exposição; foi definida a cópia dos arquivos essenciais do projeto da pasta local para o sistema que será iniciado pelo *Docker*; por fim, foi estabelecido o arquivo de inicialização das aplicações.

Com isso, para a inicialização do *Dockerfile*, foi elaborado um *docker-compose.yml* no qual são especificados diversos parâmetros essenciais para o correto funcionamento desse sistema. Entre eles estão: a configuração dos volumes, o mapeamento das portas e os protocolos destinados ao servidor DNS e, adicionalmente, são definidas as políticas de reinicialização do sistema, os métodos de montagem e os locais de montagem da imagem. Essas configurações garantem a execução adequada e eficiente do ambiente *Docker*.

4. Resultados

Nesta seção serão expostos os resultados obtidos e as análises desses dados.

4.1. Taxas de Detecção

Os testes presentes nesta seção compreendem o escopo de eficiência na detecção de domínios em relação aos dados presentes no banco, considerando a verificação da taxa de falsos positivos detectados.

Para gerar esses dados foram efetuados dois testes sintéticos, um com exatas 100 requisições e outro com 1000 requisições, em que ambos os testes possuem 10% de domínios maliciosos e um número variável de domínios que possuem relatos maliciosos porém utilizam domínios oficiais da Google ou Microsoft para hospedar alguma forma de *phishing*.

4.1.1. Teste com 100 requisições

Na Figura 10 são apresentados os resultados provenientes do teste consistindo em 100 requisições. Observa-se que 2% das requisições feitas a domínios associados ao Google e a Microsoft foram erroneamente identificadas como maliciosas, apesar de possivelmente não o serem. Por outro lado, é importante ressaltar que todas as requisições a domínios autenticamente maliciosos, que representam 10% do total, foram identificadas com precisão (sinalizados com ≪phishing≫).

```
------------------------------------------------------------
    <phishing>attdkjdposervice12.weeblysite.com
    <phishing>www.smbec-caserd.co.jp.y725.top
    <phishing>messagerie68.godaddysites.com
    <phishing>ipfs.eth.aragon.network
    forms.office.com
    <phishing>dell-com-viewer.firebaseapp.com
    <phishing>rv0ld9.webwave.dev
    <phishing>hsbc-deviceapproval.com
    docs.google.com
    <phishing>ANY.aeonkv.com
    <phishing>banrraurls.web.app
    <phishing>janusz.denisbiernacki.pl
------------------------------------------------------------
```

Figura 10: Teste de 100 requisições.

O sistema, ao verificar os domínios *docs.google.com* e *forms.office.com*, acabou relacionando as seguintes URLs maliciosas hospedadas de forma legítima nesses domínios.

```
https://docs.google.com/presentation/d/e/2PACX-1vRwu-JgwOMxtEHBwqWMnPtQiH7UH4AkCnwFa5Yx...
https://forms.office.com/pages/responsepage.aspx?id=DQSIkWdsW0yxEjajBLZtrQAAAAAAAAAAAAO...
```

Com isso, é necessário que o gerente da rede verifique os logs e trate justamente ao usuário da máquina que efetuou o acesso, para prevenir possíveis problemas relacionados a *phishing*.

4.1.2. Teste com 1000 requisições

Na Figura 11, é apresentado os resultados decorrentes de um teste composto por 1000 requisições. Observa-se um cenário semelhante ao teste anterior, no qual os domínios Google e Microsoft são suscetíveis a falsos positivos, devido a sua alta frequência de acesso.

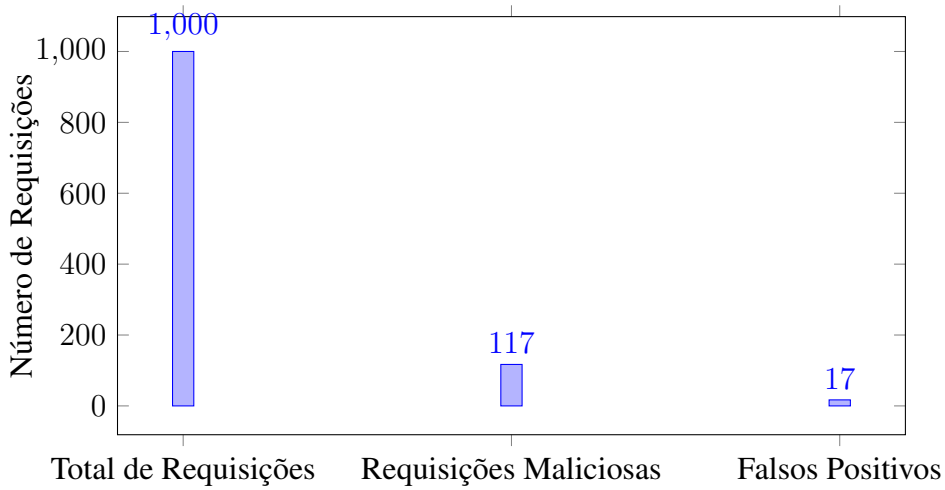

Figura 11: Teste de 1000 requisições.

4.1.3. Teste de wildcard de domínios

Considerando que vários subdomínios, por exemplo 'docs.google.com', podem ser mapeados para um único domínio utilizando *wildcard* [wil 2023], por exemplo 'google.com'. Com isso, este teste irá analisar a detecção de subdomínios e domínios maliciosos.

No exemplo abaixo, podemos observar a detecção de um subdomínio malicioso presente no banco de dados, denominado 'gateway.ipfs.io'. Quando esse subdomínio é consultado (Figura 12 - 1), o sistema o identifica como malicioso (Figura 12 - 4). Contudo, ao realizar a busca pelo domínio 'ipfs.io' (Figura 12 - 2), ele também é reconhecido como malicioso (Figura 12 - 5), uma vez que está contido na consulta original do banco de dados, que é 'gateway.ipfs.io'. Por outro lado, ao realizar uma consulta que não está presente nas entradas originais do banco de dados, como no caso da *query* 'asd.gateway.ipfs.io' (Figura 12 - 3), ela não é reconhecida como maliciosa.

```
------------------------------------------------------------
Registro de dominios do servidor DNS:
1:
08-Nov-2023 20:42:03.038 queries: info: client @0xb5a5e3f4 192.168.3.9#33311
(gateway.ipfs.io): query: gateway.ipfs.io IN A +E(0)K (172.18.0.2)
2:
08-Nov-2023 20:42:06.707 queries: info: client @0xb5a55344 192.168.3.9#49379
(ipfs.io): query: ipfs.io IN A +E(0)K (172.18.0.2)
```

```
3:
08-Nov-2023 20:42:11.187 queries: info: client @0xb5a54424 192.168.3.9#59885
(asd.gateway.ipfs.io): query: asd.gateway.ipfs.io IN A +E(0)K (172.18.0.2)

-----------------------------------------------------------
Registro de dominios maliciosos detectados pelo sistema:
4:
DNS query:08-Nov-2023 20:42:03.038 queries: info: client @0xb5a5e3f4 192.168.3.9
#33311
(gateway.ipfs.io): query: gateway.ipfs.io IN A +E(0)K (172.18.0.2), Phishing URL:
https://gateway.ipfs.io/ipfs/bafkreid4ovxck26zybwzumxngpchys4p7omdcwcz5feptm3xry
5tkctp2i
5:
DNS query:08-Nov-2023 20:42:06.707 queries: info: client @0xb5a55344 192.168.3.9
#49379
(ipfs.io): query: ipfs.io IN A +E(0)K (172.18.0.2), Phishing URL:https://gateway.
ipfs.io/ipfs/bafkreid4ovxck26zybwzumxngpchys4p7omdcwcz5feptm3xry5tkctp2i
-----------------------------------------------------------
```

Figura 12: Teste de wildcard.

4.1.4. Problemas

Os problemas encontrados durante todo o processo de desenvolvimento e testes deste projeto foram:

Por ser um sistema de detecção de domínios maliciosos e para isso foi necessário utilizar um banco de dados de URLs maliciosas, abre uma margem de erro nesta verificação por conta que se um domínio pode possuir várias URLs saudáveis e uma delas maliciosa isso não o torna um domínio malicioso, mas sim diminui o nível de confiança nesse domínio. Com isso, nota-se a necessidade de alteração do banco de dados de amostras maliciosas e a categorização dos domínios.

Por possuir uma infraestrutura limitada, foi necessário efetuar testes sintéticos obtendo, dessa forma, uma hipótese de comportamento em vida real. Com isso, seria de maior benefício analítico para os testes se eles fossem feitos em uma infraestrutura devidamente preparada para esse tipo de sistema.

5. Considerações finais

Neste trabalho foi realizada uma demonstração dos procedimentos e dos componentes empregados na criação de uma aplicação dedicada à detecção de domínios associados a ataques de *phishing*, ao mesmo tempo que oferece funcionalidades para a resolução de solicitações DNS. O objetivo primordial deste trabalho é contribuir com pesquisas na área de segurança cibernética, visando a mitigação dessa ameaça que perdura na sociedade.

É crucial observar que devido à natureza desta implementação, a qual se baseia nos domínios dos sites como ponto de partida, existe a possibilidade de ocorrer uma taxa de falsos positivos, como ilustrado na Seção 4. Diante desse cenário, recai sobre o administrador de rede a responsabilidade de lidar com esse tipo de incidente, demandando uma abordagem proativa e eficiente na resolução desses problemas. Portanto, é essencial que os profissionais envolvidos estejam preparados para lidar com essas situações com destreza e discernimento, a fim de manter a integridade e a segurança da rede.

Como aperfeiçoamento para trabalhos futuros, é possível ressaltar o problema com verificações de domínios que pode causar falsos positivos, podendo ser adicionada uma lista de domínios a serem desconsiderados nas verificações. Também é necessário efetuar mais testes para verificar o desempenho do sistema como um todo, tanto ao responder requisições quanto na análise dos domínios resolvidos.

Referências

(2023). What is wildcard dns record?

Aleroud, A. and Zhou, L. (2017). Phishing environments, techniques, and countermeasures: A survey. *Computers & Security*, 68:160–196.

Chiew, K. L., Yong, K. S. C., and Tan, C. L. (2018). A survey of phishing attacks: Their types, vectors and technical approaches. *Expert Systems with Applications*, 106:1–20.

da Silva, C. M. R., Feitosa, E. L., and Garcia, V. C. (2020). Heuristic-based strategy for phishing prediction: A survey of url-based approach. *Computers & Security*, 88.

de Almeida, R. A. O. C. B. (2022). Heuristic phishing detection based on web crawling and user behaviour monitoring with a deterministic approach for cybersecurity.

Docker, I. (2023). Docker overview. Disponível em: `https://docs.docker.com/get-started/overview/`. Acessado em Setembro de 2023.

Internet Systems Consortium, I. (2023). Bind 9 administrator reference manual. Disponível em: `https://bind9.readthedocs.io/en/v9.18.14/chapter1.html`. Acessado em Setembro de 2023.

James, L. (2006). Chapter 1 - banking on phishing. In James, L., editor, *Phishing Exposed*, pages 1–35. Syngress, Burlington.

Montagner, A. S. and Westphall, C. M. (2022). Uma breve análise sobre phishing. *Revista ComInG-Communications and Innovations Gazette*, 6(1):46–56.

Network Working Group, I. (1987). Domain names - concepts and facilities. Disponível em: `https://www.ietf.org/rfc/rfc1034.txt`. Acessado em Setembro de 2023.

Python Software Foundation, I. (2022). What is python? executive summary. Disponível em: `https://www.python.org/doc/essays/blurb/`. Acessado em Setembro de 2023.

SolidRun (2017). Cubox-i – the little computer that can. Disponível em: `https://www.solid-run.com/wiki/lib/exe/fetch.php?media=imx6:cubox-i:brochure_imx6_cubox_2017-09-05.pdf`. Acessado em Setembro de 2023.

Systems, C. (2022). Phishtank. Disponível em: `https://phishtank.org/`. Acessado em Setembro de 2023.