COMMUNICATIONS AND INNOVATIONS GAZETTE

# 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].

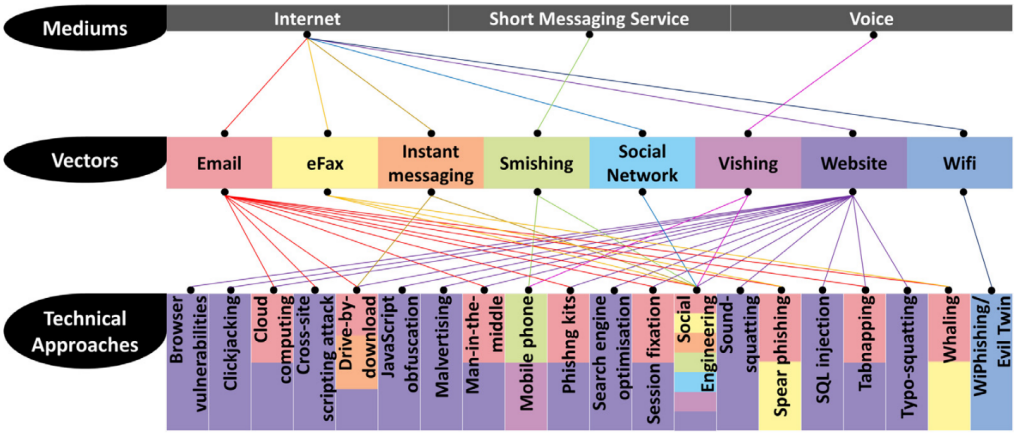


Figura 1: A interligação entre meio, vetor e abordagem das técnicas de *phishing*. [Chiew et al. 2018]

**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.

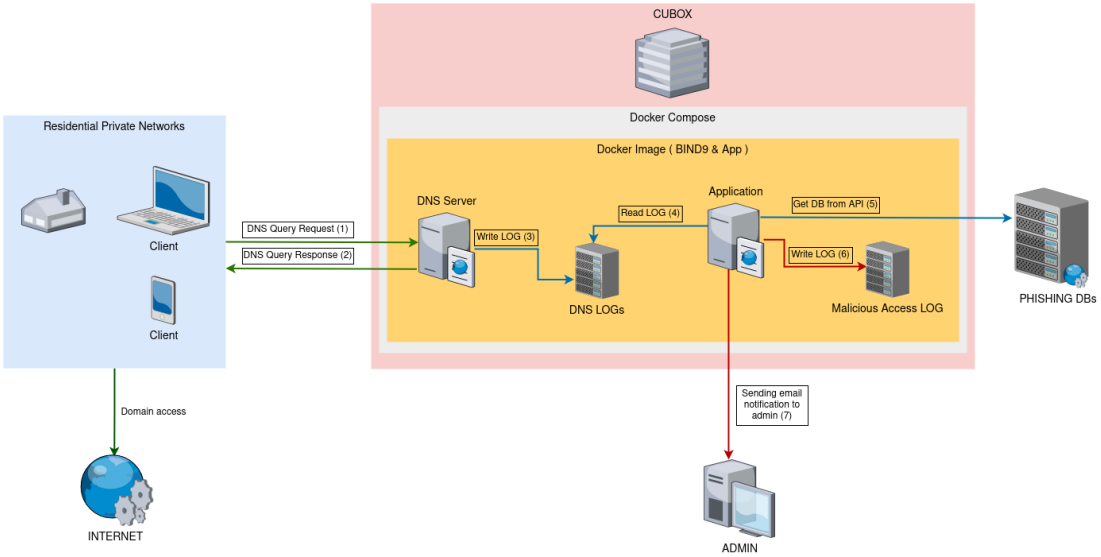


Figura 2: Topologia do sistema desenvolvido.

### 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.

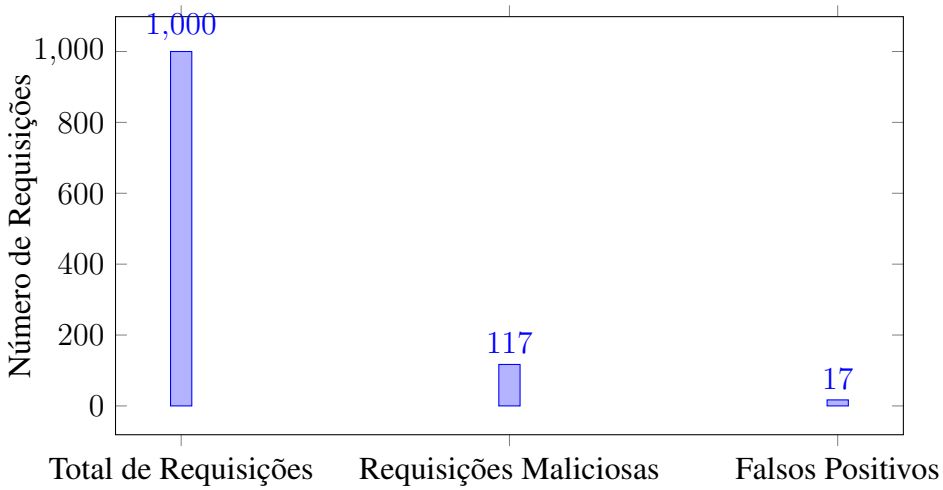


Figura 11: Teste de 1000 requisições.

### 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.