UNIVERSIDADE NOVE DE JULHO – UNINOVE
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA E GESTÃO DO CONHECIMENTO

JOÃO RAFAEL GONÇALVES EVANGELISTA

ABORDAGEM DE INTELIGÊNCIA DE FONTES ABERTAS COM MAPAS AUTO-ORGANIZÁVEIS DE KOHONEN E PROCESSAMENTO DE LINGUAGEM NATURAL PARA EXECUÇÃO AUTOMÁTICA DE DORKS

São Paulo
2020

JOÃO RAFAEL GONÇALVES EVANGELISTA

ABORDAGEM DE INTELIGÊNCIA DE FONTES ABERTAS COM MAPAS AUTO-ORGANIZÁVEIS DE KOHONEN E PROCESSAMENTO DE LINGUAGEM NATURAL PARA EXECUÇÃO AUTOMÁTICA DE DORKS

Texto de Defesa apresentado ao Programa de Pós-Graduação em informática e Gestão do Conhecimento da Universidade Nove de Julho - UNINOVE, como requisito parcial para a obtenção do título de Mestre em Informática e Gestão do Conhecimento.

Linha de Pesquisa 3: Tecnologia da Informação e Conhecimento.

Orientador: Prof. Dr. Renato José Sassi

São Paulo

2020

Evangelista, João Rafael Gonçalves.
Abordagem de inteligência de fontes abertas com mapas auto-organizáveis de Kohonen e processamento de linguagem natural para execução automática de dorks. / João Rafael Gonçalves Evangelista. 2020.
155 f.
Dissertação (Mestrado) – Universidade Nove de Julho - UNINOVE, São Paulo, 2020.
Orientador (a): Prof. Dr. Renato José Sassi.
1. OSINT. 2. Inteligência de fontes abertas. 3. Dorks. 4. Processamento de linguagem natural. 5. Mapas auto-organizáveis de Kohonen.
I. Sassi, Renato José. II. Título.
CDU 004

UNINOVE
Universidade Nove de Julho

PARECER – EXAME DE DEFESA

Parecer da Comissão Examinadora designada para o exame de defesa do Programa de Pós-Graduação em Informática e Gestão do Conhecimento a qual se submeteu o aluno João Rafael Gonçalves Evangelista.

Tendo examinado o trabalho apresentado para obtenção do título de "Mestre em Informática e Gestão do Conhecimento", com Dissertação intitulada "ABORDAGEM DE INTELIGÊNCIA DE FONTES ABERTAS COM MAPAS AUTO-ORGANIZÁVEIS DE KOHONEN
E PROCESSAMENTO DE LINGUAGEM NATURAL PARA EXECUÇÃO AUTOMÁTICA DE DORKS", a Comissão Examinadora considerou o trabalho:

(X) Aprovado **( ) Aprovado condicionalmente**
( ) Reprovado com direito a novo exame **( ) Reprovado**

Parecer:
Aprovado_____
_____.

EXAMINADORES

Prof. Dr. Renato José Sassi (Orientador – UNINOVE)
_____

Prof. Dr. Marcirio Silveira Chaves (Membro Externo – PUC/RS)
_____

Prof. Dr. Domingos Marcio Rodrigues Napolitano (Membro Interno – UNINOVE)
_____

Prof. Dr. Peterson Adriano Belan (Membro Interno – UNINOVE)
_____

São Paulo, 29 de junho de 2020

Dedico este trabalho à minha companheira, meus pais, dedico também ao Prof. Dr. Renato José Sassi, meu orientador. Vocês representaram a motivação central para o desenvolvimento e a conclusão deste trabalho.

AGRADECIMENTOS

Agradeço, em primeiro lugar, a Deus, que sempre esteve ao meu lado, orientando os meus passos, e colocando pessoas maravilhosas em meu caminho.

À Universidade Nove de Julho (UNINOVE) pelo apoio e pela oportunidade de crescimento e aprimoramento acadêmico, pessoal e profissional, assim como pela bolsa de estudos.

À minha companheira, Mayara Thábata Sá de Lima, que sempre me apoiou e incentivou a jamais desistir dos meus objetivos.

Aos meus pais, Edinalva Gonçalves e José Elizeu Evangelista, que sempre me incentivaram a estudar e me mostrarm como a educação transforma as pessoas.

Ao Centro Paula Souza e aos meus colegas de trabalho, em especial, Maria Inês, Messias e Tarsila que me possibilitaram desenvolver este trabalho.

Aos professores e colegas de universidade que me auxiliaram de maneira direta ou indireta. Em especial, aos meus colegas de pesquisa do PPGI, Domingos, Ricardo, Andréia, Rosana, Ângelo, Pâmela, Edquel e Marcio Romero, pelas dicas, pelo apoio e pelos conselhos ao longo desta jornada que é a vida acadêmica.

Ao Prof. Dr. André Felipe Henriques Librantz que, juntamente com meu orientador, Prof. Dr. Renato José Sassi, acreditaram no meu potencial e me proporcionaram momentos de crescimento e aprendizado.

Ao meu orientador, Prof. Dr. Renato José Sassi, pelo apoio, suporte e conhecimento, e pela confiança, paciência, coordenação e disponibilidade.

Enfim, os meus sinceros agradecimentos a todos que de alguma forma contribuíram para a minha jornada acadêmica.

"As coisas que já imaginei que seriam minhas maiores conquistas foram só os primeiros passos rumo a um futuro que começo, meramente, a vislumbrar."

Jace Beleren

RESUMO

Para proteger as informações localizadas na internet, a área da Segurança da Informação dispõe de um processo para testar a segurança de páginas web, denominado Teste de Intrusão ou Pentest. Em sua fase inicial, o teste visa realizar buscas na internet a fim de reunir o máximo de informações disponíveis para apoiar as demais fases do processo e, até mesmo, já encontrar algumas vulnerabilidades. Essa fase inicial é chamada de Inteligência de Fontes Abertas, ou, em inglês, *Open Source Intelligence* (OSINT). Uma prática para OSINT utilizada em Pentest é o Google Hacking, que funciona aplicando strings denominadas Dorks. O Google Hacking pode ser executado de duas formas: manual e automática, sendo que a primeira possui um tempo de duração superior ao da segunda. Uma maneira de melhorar o desempenho do Pentest é torná-lo automático com a aplicação de técnicas de Inteligência Artificial (IA), como os Mapas Auto-Organizáveis (SOM) de Kohonen, um tipo de rede neural artificial utilizada para gerar agrupamentos, e o Processamento de Linguagem Natural (PLN), uma subárea da IA responsável por fazer com que os computadores interpretem e desenvolvam conteúdo em linguagem humana. Assim, o objetivo deste trabalho foi desenvolver uma abordagem de Inteligência de Fontes Abertas, por meio dos Mapas Auto-Organizáveis de Kohonen e do Processamento de Linguagem Natural, para execução automática de Dorks, a fim de melhorar o desempenho da prática do Google Hacking. A base de dados selecionada foi o Google Hacking Database (GHDB), contendo 4.211 Dorks e 4 atributos. A abordagem proposta neste trabalho foi desenvolvida em 10 fases: preparação do ambiente para executar o OSINT, definição do escopo de OSINT, seleção da base de Dorks, seleção e avaliação das ferramentas OSINT, pré-processamento da base de Dorks, transformação da base de Dorks, aplicação da rede SOM na base de Dorks, análise dos resultados, adição das novas informações na base de Dorks e validação da execução automática de Dorks. Os resultados obtidos apontaram um melhor desempenho da abordagem proposta quando executada automaticamente a base de Dorks comparada à execução manual. Desta forma, concluiu-se que a abordagem de Inteligência de Fontes Abertas, utilizando Mapas Auto-Organizáveis de Kohonen e Processamento de Linguagem Natural, pode ser aplicada na execução automática de Dorks.

Palavras-chave: OSINT, Inteligência de Fontes Abertas, Dorks, Processamento de Linguagem Natural, Mapas Auto-Organizáveis de Kohonen, Pentest Automático

ABSTRACT

To protect the information located on the Internet, the Information Security area has a process to test the security of web pages, called Intrusion Test or Pentest. In its initial phase, the test performs searches on Web pages to gather or obtain the maximum information available to support other phases of the process, or even, find some vulnerabilities. This phase is called Open Source Intelligence (OSINT). A practice for OSINT used in Pentest is Google Hacking, which works using *strings* called Dorks. Google Hacking can be performed in two ways: manual and automatic, and the first has a longer duration than the second. One way to improve Pentest performance is to make it automatic with the application of Artificial Intelligence (AI) techniques, such as Kohonen's Self-Organizing Maps (SOM), a type of artificial neural network used to generate clusters, and Natural Language Processing (PLN), a subarea of AI responsible for making computers able to interpret and develop content in human language. So, the objective of this work was to develop an Open Source Intelligence Approach with Kohonen's Self-Organizing Maps and Natural Language Processing for automatic execution of Dorks to improve the performance of the Google Hacking practice. The selected database was the Google Hacking Database (GHDB), containing 4,211 Dorks and 4 attributes. The approach proposed in this work was developed in 10 phases: preparing the environment to run OSINT, defining the OSINT scope, selecting the Dorks base, selecting and evaluating the OSINT tools, pre-processing the Dorks base, transforming the Dorks base, application of SOM in the Dorks base, analysis of results, addition of new information in the Dorks database and validation of the automatic execution of Dorks. The results obtained revealed a better performance of the proposed approach when automatically executing the Dorks base compared to manual execution. It was concluded, then, that Open Source Intelligence Approach with Kohonen's Self-Organizing Maps and Natural Language Processing can be applied in the automatic execution of Dorks.

Keywords: OSINT, Open Source Intelligence, Dorks, Natural Language Processing, Self-Organizing Maps, Automatic Pentest.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

ABNT Associação Brasileira De Normas Técnicas

ASCII *American Standard Code for Information Interchange* (Código Padrão Americano para o Intercâmbio de Informação)

BMU *Best Match Unit* (Neurônio Vencedor)

CIA *Central Intelligence Agency* (Agência Central de Inteligência)

CIP *Critical Infrastructure Protection* (Proteção De Infraestruturas Críticas)

COBIT *Control Objectives For Information And Related Technology* (Objetivos De Controle Para Tecnologia Da Informação E Áreas Relacionadas)

CTI *Cyber Threat Intelligence* (Inteligência De Ameaças Cibernéticas)

DOD *Department Of Defense* (Departamento De Defesa)

EQ Erro De Quantização

ET Erro Topográfico

EUROPOL Europe Police Services (Serviços Poiliciais da Europa)

FBI *Federal Bureau of Investigation* (Departamento Federal de Investigação)

GHDB *Google Hacking Database* (Base de Dados *Google Hacking*)

HTML *HyperText Markup Language* (Linguagem de Marcação de HiperTexto)

IA Inteligência Artificial

IEEE *Institute Of Electrical And Electronic Engineers* (Instituto De Engenheiros Eletricistas E Eletrônicos)

ISECOM *The Institute For Security And Open Methodologies* (Instituto De Segurança E Metodologias Abertas)

ISO *International Organization For Standardization* (Organização Internacional Para Padronização)

ISSAF *Information Systems Security Assessment Framework* (Estrutura De Avaliação De Segurança De Sistemas De Informação)

ITIL *Information Technology Infrastructure Library* (Biblioteca De Infraestrutura De Tecnologia Da Informação)

MLP *Multilayer Perceptron* (Perceptron Multicamadas)

NERC *North American Electric Reliability Corporation* (Corporação Norte-Americana De Confiabilidade Elétrica)

NIST *National Institute Of Standards And Technology* (Instituto Nacional de Padrões e Tecnologia)

OTAN Organização do Tratado do Atlântico Norte

OISSG *Open Information Security Service Group* (Grupo de Serviço de Segurança da Informação Aberta)

OSINT *Open Source Intelligence* (Inteligência de Fontes Abertas)

OSSTMM *The Open Source Testing Methodology Manual* (Manual de Metodologia de Teste de Código Aberto)

OWASP *The Open Web Application Security Project* (Projeto de segurança de Aplicações Abertas Web)

PENTEST *Penetration Test* (Teste de Intrusão)

PLN Processamento De Linguagem Natural

PTES *The Penetration Testing Execution Standard* (Padrão de Execução de Testes de Intrusão)

RNA Redes Neurais Articiais

RSL Revisão Sistemática da Literatura

SI Segurança da Informação

SOM *Self-Organizing Maps* (Mapas Auto-Organizáveis de Kohonen)

SQL *Structured Query Language* (Linguagem de Consulta Estruturada)

URL *Uniform Resource Locator* (Localizador Padrão de Recursos)

SUMÁRIO

1 INTRODUÇÃO

Segundo Ly *et al*. (2018), na era digital, as pessoas conseguem acessar a internet por meio de inúmeros dispositivos, permitindo-as buscar informações diariamente. Além disso, o acesso à internet possibilita que as pessoas compartilhem informações de diversos tipos.

Embora o compartilhamento de informações na internet traga uma série de vantagens, tal ação também traz desvantagens, pois afeta diretamente na preservação da privacidade, uma vez que informações compartilhadas podem ser acessadas de forma indevida (MEDKOVA, 2018).

Práticas como a publicação de listas de funcionários, projetos, eventos e até mesmo de fornecedores em páginas web organizacionais permitem que pessoas mal-intencionadas identifiquem facilmente informações sobre os colaboradores de uma empresa entre milhões de usuários de mídias sociais (EDWARDS *et al.*, 2017).

Organizações governamentais, empresas e o público de forma geral contribuem para o aumento de informações compartilhadas na internet. Do mesmo modo que o volume de informações na internet cresce exponencialmente, o número de novas ameaças também aumenta, dificultando a proteção das informações (NAARTTIJÄRVI, 2018).

Para garantir a proteção das informações compartilhadas na internet ou em outros sistemas de informação, existe a área da Segurança da Informação (SI). Segundo a norma ISO 17799:2005, a segurança da informação é a área responsável pela proteção de informações de vários tipos de ameaças, para garantir a continuidade do negócio, minimizar o risco ao negócio e maximizar o retorno sobre os investimentos e as oportunidades de negócio.

Quanto às propriedades da informação que deve ser protegida, a norma ISO 27001:2006 define que a segurança da informação envolve a preservação de sua confidencialidade, integridade e disponibilidade; adicionalmente, de outras propriedades, como a autenticidade e a legalidade.

A crescente dependência das organizações e das pessoas em geral pela segurança da informação fez surgir padrões, métodos, serviços e tecnologias com o objetivo de auxiliar a gestão e a prática da área (HAUFE *et al.,* 2016; HAQAF, KOYUNCU, 2018).

Como exemplo de padrões para a segurança da informação, pode-se citar a série ISO 27000:2016, um conjunto de normas responsáveis por auxiliar a execução e manutenção das

melhores práticas para a área. Quanto aos métodos, pode-se citar a criação de políticas de segurança da informação para adequar uma organização às leis vigentes de seu país. Já no que se refere a serviços e tecnologia, existe o desenvolvimento de softwares para criação de cópias de segurança de arquivos (backups), a fim de diminuir o risco de perdê-los.

Uma forma de garantir a segurança da informação é descobrir as vulnerabilidades existentes no lugar em que a informação está armazenada, como em uma página web, por exemplo. As vulnerabilidades representam falhas de segurança, que são capazes de proporcionar riscos para a informação e sua entidade proprietária, seja organização ou pessoa (DOBROVOLJC; TRČEK; LIKAR, 2017).

As vulnerabilidades podem ser analisadas de duas formas: pelo método de avaliação de vulnerabilidades e pelo processo de teste de intrusão. A avaliação de vulnerabilidade é um método utilizado para analisar ativos ou sistemas de informação com o objetivo de encontrar vulnerabilidades, mas sem explorá-las (NAGPURE; KURKURE, 2017).

Segundo Hatfield (2019), o teste de intrusão, do inglês, *Penetration Test* (Pentest), é o processo responsável por encontrar e explorar vulnerabilidades, para assim garantir a segurança de computadores, sistemas ou redes por meio de simulações reais de invasão. A diferença entre o Pentest e uma intrusão é que o primeiro é realizado de acordo com um contrato pré-estabelecido, enquanto o segundo é um ato ilícito classificado como crime cibernético por infringir as propriedades da informação definida pela ISO 27001:2006.

Na fase preparatória do Pentest, conhecida como “Reconhecimento”, realizam-se buscas em páginas web ou em outras fontes abertas para reunir o máximo de informações disponíveis sobre um determinado alvo, podendo assim elaborar o escopo do Pentest e até mesmo já encontrar algumas vulnerabilidades (ROY *et al.*, 2017).

O conceito que envolve a coleta, a análise e o uso de informações de fontes abertas, como as páginas web, para propósitos inteligentes, ou seja, para gerar conhecimento que possa ser utilizado em algum objetivo pré-determinado, é chamado de Inteligência de Fontes Abertas, ou *Open Source Intelligence* (OSINT), em inglês (KOOPS, HEOPMAN E LEENES; 2013).

OSINT permite coletar e analisar informações de diversos tipos e tamanhos, disponíveis em fontes abertas, como a internet, tornando possível a realização de auditorias em sistemas e testes de segurança da informação, como o Pentest (RICO *et al.*, 2018).

Para executar OSINT em Pentest, são utilizados ferramentas, *framework*s e abordagens OSINT, a fim de facilitar a busca e coleta de informações. Entre suas características, destaca-se a capacidade de selecionar e organizar as informações encontradas de forma automática. Uma das principais ferramentas utilizadas para realizar o OSINT é o mecanismo de busca do Google (CHEN; DÉCARY, 2018).

O Google é um mecanismo de busca capaz de procurar informações na internet utilizando operadores de busca e sistemas de cache. A prática de utilizar o Google em Pentest é denominada Google Hacking, o qual funciona com *strings* específicas compostas por operadores de pesquisa do Google. As *strings*, também chamadas de Dorks, são sequências de caracteres utilizados para realizar uma busca específica no Google (ROY *et al.*, 2017).

O objetivo do Google Hacking não é encontrar informações genéricas, como o Google geralmente faz, mas sim encontrar informações que auxiliem na execução de um Pentest. Utilizam-se as Dorks para diferentes fins, como encontrar vulnerabilidades na estrutura de um site, em arquivos de banco de dados expostos, em *logs* de serviços ativos e em arquivos infectados com vírus (TOFFALINI *et al*., 2016).

Google Hacking Database (GHDB) é uma base que contém Dorks testadas e validadas pela Offensive Security, uma organização especializada em segurança da informação. Trata-se da principal base mundial para aplicação da prática do Google Hacking em Pentest. As Dorks presentes na base GHDB são categorizadas conforme sua atuação, pois cada Dork possui uma categoria (TOFFALINI *et al*., 2016).

O Pentest pode ser executado de duas formas: automático e manual. A execução manual ainda é a mais popular devido à variedade de vulnerabilidades existentes em sistemas de informação. Por outro lado, é mais fácil de ser detectada por sistemas de defesa, além de ser mais lenta comparada ao Pentest automático. Por muitas vezes, faz-se necessário que um avaliador acompanhe o Pentest para revisar o que foi feito e classificar corretamente as informações utilizadas e as que foram encontradas (TANG, 2014).

Para melhorar o desempenho do Pentest e de suas práticas, uma solução encontrada por profissionais de segurança da informação foi desenvolver algoritmos, ou mesmo sistemas que automatizam algumas fases do teste, a fim de executar o Pentest mais rapidamente (STEFINKO; PISKOZUB; BANAKH, 2016).

Para a fase preparatória do Pentest, em que o OSINT é executado, Noubours, Pritzkau e Schade (2014) abordam a aplicação de Inteligência Artificial (IA) e de sua subárea, o Processamento de Linguagem Natural (PLN) para obter ganhos de desempenho e conseguir extrair informações dos resultados obtidos. A Inteligência Artificial (IA) é a área da ciência da computação responsável por desenvolver técnicas que permitem que computadores resolvam problemas complexos e, assim, auxiliem na tomada de decisão (MCKINNEL *et al*., 2019).

Segundo Wang, Lu e Qin (2020), algumas técnicas de IA podem ser aplicadas na área de segurança da informação. As Redes Neurais Artificiais (RNA), por exemplo, podem ser usadas para controlar e análisar o tráfego de rede, possibilitando a detecção de ameaças e anomalias, na filtragem de spam e *phishings* na análise de dados históricos, como *logs* de softwares e sistemas de defesa.

Um tipo de arquitetura de RNA que pode ser utilizada na área da segurança da informação são os Mapas Auto-Organizáveis de Kohonen, ou *Self-Organizing Maps* (SOM), em inglês (López *et al*., 2019). Segundo Kohonen (1997), a rede SOM é capaz de extrair conhecimento de base de dados, levando em consideração todos os seus atributos de forma simultânea e formando agrupamentos por similaridade.

Essa capacidade permite que a rede SOM seja aplicada na área de segurança da informação, para investigar evidências digitais em computadores (BELLA; HELLOF, 2016) e detectar anomalias em ambientes on-line (LEE; KIM; KIM, 2011).

Já o Processamento de Linguagem Natural (PLN), área conhecida também como linguística computacional, envolve o aprendizado, a compreensão, o reconhecimento e a produção de conteúdo em linguagem humana por sistemas computacionais (ZEROUAL; LAKHOUAJA, 2018).

Segundo Noubours, Pritzkau e Schade (2014), a aplicação de PLN em segurança da informação, especificamente em OSINT, é uma forma de aumentar o desempenho das ferramentas utilizadas e, consequentemente, melhorar o desempenho na descoberta de vulnerabilidades, principalmente as que já são conhecidas e documentadas.

Diante do exposto, considera-se relevante desenvolver uma abordagem de Inteligência de Fontes Abertas com Mapas Auto-Organizáveis de Kohonen e Processamento de Linguagem

Natural, para execução automática de Dorks, a fim de melhorar o desempenho da prática do Google Hacking.

1.1 JUSTIFICATIVA E MOTIVAÇÃO

A informação é um dos bens mais importantes para a organização, bem como a necessidade de garantir a sua segurança. Esta imposição por proteção para as informações já serve de justificativa para a realização deste trabalho (MEDKOVA, 2018).

No entanto, problemas de segurança da informação, como a perda da confidencialidade ou integridade de informações, localizadas em páginas de web empresariais, podem afetar as finanças e a reputação de uma empresa (Edwards *et al.*, 2017). Sendo assim, considera-se importante desenvolver novas abordagens para testar as páginas web, a fim de encontrar suas vulnerabilidades e corrigi-las.

O tema segurança da informação vem sendo constantemente pesquisado pela comunidade acadêmica, uma vez que as ameaças vêm aumentando consideravelmente e tornando-se cada vez mais complexas, o que abre espaço para o desenvolvimento e a aplicação de abordagens que utilizem técnicas de IA.

A aplicação de PLN e SOM corrobora com a afirmação acima e vem ao encontro de uma motivação que é tendência mundial, ou seja, a utilização de IA para resolver problemas de diversas áreas, principalmente da ciência da computação.

A revisão sistemática da literatura realizada, que pode ser encontrada no Capítulo 3 deste trabalho, não encontrou nenhum estudo que tenha aplicado a rede SOM ou outra arquitetura de RNA para OSINT. Apenas contatou-se sua aplicação em outras fases do Pentest, como, por exemplo, na análise dos resultados e na construção de relatórios (MCKINNEL *et al*., 2019).

No caso do PLN, encontrou-se o estudo de Nouburs, Pritzkau e Schade (2014), que descreveram diversas aplicações de PLN em um *framework* OSINT, confirmando assim um aumento em seu desempenho. No entanto, não se evidenciou na revisão sistemática da literatura realizada a aplicação de PLN em Dorks para melhorar o desempenho do Google Hacking.

No que tange à execução do Pentest, Chu e Lisitsa (2018), Nagpure e Kurkure (2017) e Hatfield (2018) afirmam que a forma manual ainda é a mais utilizada, inclusive em sua fase

inicial, chamada de “Reconhecimento” ou OSINT. Os autores afirmam também que a execução do Pentest de forma manual leva mais tempo do que a execução automática, e o tempo é um fator crucial para a detecção de vulnerabilidades. Assim, quanto mais cedo as vulnerabilidades forem detectadas e corrigidas, menor será o risco de as informações serem comprometidas, o que abre espaço para estudos sobre execução automática.

Em seu estudo, Bae, Lim e Cho (2016) executaram Dorks do Google Hacking Database de forma automática com o software SiteDigger3, e concluíram que tal programa possui limitações na quantidade de Dorks a serem executadas e na inserção de novas Dorks na ferramenta. Assim, os autores recomendam pesquisar outros meios de execução automática, o que possibilita o desenvolvimento de novas abordagens sobre o assunto.

Considera-se então que, para a fase inicial do Pentest, em que OSINT é aplicado, a execução automática de Dorks é fundamental para descobrir vulnerabilidades e informações que podem levar até elas de forma mais rápida.

Por fim, não se encontrou nenhum estudo que abordasse a aplicação da rede SOM e PLN em Dorks para o Google Hacking, ou mesmo em outra prática utilizada em um Pentest, a fim de torná-la automático.

Desta forma, tem-se aqui justificado o desenvolvimento de uma abordagem reunindo as técnicas SOM e PLN, sendo que a rede SOM pode ser aplicada em uma base de Dorks para extrair conhecimento ao formar agrupamentos de Dorks por similaridade para serem executados de forma automática, e o PLN pode ser aplicado para pré-processar a base de Dorks com o objetivo de prepará-la para a aplicação da SOM.

Assim, vislumbra-se a contribuição deste trabalho para a academia, ao desenvolver uma abordagem de OSINT com PLN e SOM para executar Dorks da base GHDB de forma automática. Em adição, a abordagem desenvolvida não só possibilita a execução automática da prática do Google Hacking para encontrar vulnerabilidades, como também possibilita a realização de agrupamentos de palavras ou sentenças por similaridade em seus caracteres.

Outra contribuição considerada é a revisão sistemática da literatura realizada, em que foram encontradas 248 publicações sobre OSINT, o que servirá de base para a consulta de pesquisadores interessados no assunto aqui abordado.

Outro ponto de destaque foi a avaliação de ferramentas para Pentest, mais especificamente ferramentas OSINT para a execução automática do Google Hacking. Essa avaliação possibilitará o desenvolvimento de novos estudos com tais ferramentas, ou mesmo o desenvolvimento de métricas para avaliação.

Para as organizações, a contribuição pode residir na utilização da abordagem para identificar vulnerabilidades de forma automática, ganhando assim mais tempo e, consequentemente, diminuindo o risco de as empresas se tornarem vítimas de crimes cibernéticos. Além disso, tal abordagem poderá apoiar a tomada de decisão de analistas e demais profissionais da área de segurança da informação.

No caso da sociedade, considera-se que este trabalho possibilitará o armazenamento dos dados pessoais de clientes e colaboradores de uma determinada organização em locais mais seguros, pois a abordagem aqui proposta pretende identificar que não há vulnerabilidades já documentadas e relatadas na base do GHDB.

Ainda sobre a sociedade, este trabalho promove a pesquisa e utilização do Pentest, uma prática de Segurança da Informação que possui o objetivo de descobrir vulnerabilidades em sistemas computacionais, para assim, inibir invasões e outros crimes cibernéticos. Desta forma, considera-se relevante esta promoção do assunto, para assegurar a conscientização de usuários para o tema de Segurança da Informação.

Vale ressaltar que a abordagem proposta para a execução automática do Google Hacking não se limita a páginas web organizacionais, podendo, portanto, ser aplicada em muitas outras, tais como: páginas web de hospitais e outras instituições de saúde, páginas de *e-commerce*, portais acadêmicos de instituições de ensino e páginas de acesso a armazenamentos em nuvem.

1.2 PROBLEMA DE PESQUISA

A velocidade com que a tecnologia evolui obriga a área da segurança da informação a acompanhar essa evolução, pois da mesma forma que as tecnologias podem ser utilizadas para o bem da sociedade, elas também podem ser usadas para fins maliciosos.

Assim, as vulnerabilidades em sistemas de informação podem fazer com que as organizações aumentem seus custos, seja contratando empresas terceirizadas para prover

segurança da informação, seja arcando com as consequências de uma vulnerabilidade explorada (FENG *et al*., 2019).

É por isso que testes de intrusão ou Pentest (*Penetration Testing*) são realizados envolvendo a simulação de ataques de hackers, para avaliar a segurança de sistemas de informação. Sua execução é feita geralmente de forma manual, seu ciclo de testes é longo quando comparado ao pentest automático. Por fim, seu resultado é um relatório que enumera as vulnerabilidades encontradas e exploradas (TETSKYI; KHARCHENKO; UZUN, 2018).

Para a área de segurança da informação, o OSINT é utilizado em Pentest para buscar vulnerabilidades ou informações que ajudem a revelá-las, sejam elas físicas, digitais, eletrônicas ou humanas (PTES TECHNICAL GUIDELINES, 2014).

A eficiência do Pentest permite que gestores, supervisores e outros profissionais de tecnologia alocados em cargos estratégicos apresentem garantias para seus colaboradores e clientes de que sua infraestrutura e seus sistemas estão atendendo os requisitos de segurança.

Desta forma, considera-se que o desenvolvimento e a aplicação da abordagem de OSINT, utilizando a rede SOM e o PLN para a execução automática de Dorks, é importante para ampliar os estudos sobre a busca de novas formas de se garantir a segurança da informação.

Tal constatação originou a seguinte questão de pesquisa: “Como desenvolver uma abordagem de Inteligência de Fontes Abertas com Mapas Auto-Organizáveis de Kohonen e Processamento de Linguagem Natural, para execução automática de Dorks, a fim de melhorar o desempenho da prática do Google Hacking? ”.

1.3 OBJETIVOS GERAL E ESPECÍFICOS

1.3.1 Objetivo Geral

Desenvolver uma abordagem de Inteligência de Fontes Abertas com Mapas Auto-Organizáveis de Kohonen e Processamento de Linguagem Natural, para execução automática de Dorks, a fim de melhorar o desempenho da prática do Google Hacking.

1.3.2 Objetivos Específicos

- Realizar uma revisão sistemática da literatura.
- Selecionar e avaliar ferramentas OSINT.
- Selecionar a base de Dorks.
- Pré-processar com PLN e converter para ASCII a base de vulnerabilidades.
- Aplicar a rede SOM para gerar agrupamentos na base de vulnerabilidades.
- Comparar o desempenho da execução automática de Dorks, realizada pela abordagem proposta, com a execução manual.

1.4 DELIMITAÇÃO DO TEMA

Os temas abordados neste trabalho estão dentro do contexto da área de segurança da informação utilizando SOM e PLN, ambas encontradas na literatura, na base GHDB para possibilitar a execução automática de Dorks.

Escolheu-se o Pentest por se tratar de um tema que vem sendo estudado por pesquisadores e profissionais da área de segurança da informação como uma possível solução para encontrar vulnerabilidades em páginas web.

Em sua fase inicial, o Pentest necessita de uma coleta de informações que suportem as suas demais fases. Nesta pesquisa, escolheu-se o Google Hacking pelo fato de essa prática conseguir extrair informações e vulnerabilidades em páginas web, diferente das demais práticas utilizadas na fase inicial do Pentest.

Quanto ao Google Hacking Database (GHDB), sua seleção se deu pelo fato de se tratar da maior e mais significativa base de Dorks on-line disponível para profissionais de segurança da informação, conforme descreve alguns estudos, como: Meucci e Muller (2014), Zhang, Notani e Gu (2015) e Mider, Garlicki e Jan (2019).

Decidiu-se por utilizar PLN pela necessidade de manipular as Dorks, que, na maioria das vezes, são compostas por caracteres especiais. Já o Mapa Auto-Organizável de Kohonen foi selecionado por se tratar de uma técnica não-supervisionada capaz de realizar agrupamentos por similaridade, permitindo assim segmentar a base GHDB e formar agrupamentos de Dorks semelhantes.

1.5 ORGANIZAÇÃO DO TRABALHO

Além deste capítulo introdutório, este trabalho está estruturado da seguinte forma:

Capítulo 2 (Fundamentação Teórica) – Neste capítulo, são apresentados os conceitos abordados no desenvolvimento deste trabalho. A saber: Segurança da Informação, Pentest, OSINT, Google Hacking, Inteligência Artificial, Aprendizagem de Máquina, Redes Neurais Artificiais, rede SOM, Processamento de Linguagem Natural.

Capítulo 3 (Materiais e Métodos) – Neste capítulo, é apresentada a metodologia utilizada para desenvolvimento do texto e da pesquisa, bem como suas características e fases.

Capítulo 4 (Apresentação e Discussão dos Resultados) – Neste capítulo, são apresentados e discutidos os resultados.

Capítulo 5 (Conclusão) – Por fim, neste capítulo, é apresentada a conclusão do presente trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, apresenta-se a fundamentação teórica dos principais temas abordados neste trabalho: Segurança da Informação, Teste de Intrusão, Inteligência de Fontes Abertas, Google Hacking, Inteligência Artificial, Mapas Auto-Organizáveis de Kohonen e Processamento de Linguagem Natural.

2.1 SEGURANÇA DA INFORMAÇÃO

Segundo a Associação Brasileira de Normas Técnicas (ABNT, 2013), a segurança da informação é a área responsável pela "proteção de ativos de informação contra acessos não autorizados, alterações indevidas ou sua indisponibilidade".

Pode-se proteger a informação de possíveis ameaças utilizando determinados recursos da área de segurança da informação, como: políticas, processos, métodos, padrões, procedimentos e ferramentas. Tais recursos auxiliam as organizações a tomarem as seguintes medidas preventivas e proativas: sistemas de defesa, redes privadas, gerenciamento de atualizações e correções, auditoria e testes de segurança (DA SILVA *et al*., 2019; FENG *et al*., 2019).

A área da segurança da informação dispõe de padrões que possuem o objetivo de fornecer as melhores práticas e técnicas para a proteção das informações e dos ativos que as mantêm. Em seu estudo, Leszczyna (2018) e Haufe *et al*., (2016) apresentam alguns padrões (Tabela 1) que podem ser utilizados para a segurança da informação.

Tabela 1 - Padrões de Segurança da Informação

Padrões	**Descrição**
ITIL	Conjunto de boas práticas para a área de tecnologia da informação.
ISO 270XX	Padrões de Segurança da Informação
ISO/ IEC 62351	Autenticação e criptografia.
ISO IEC 27000	Sistemas de gestão de segurança da informação.
NERC CIP	Proteção para infraestruturas críticas.
IEEE 1686	Dispositivos eletrônicos inteligentes
NIST SP 800-53	Segurança e controles de privacidade
NIST SP 800-82	Sistemas de controle industrial
ISO/IEC 15408	Avaliação da segurança de TI
COBIT	*Framework* para Governança da Tecnologia da Informação

Fonte: Adaptado de Leszczyna (2018) e Haufe *et al*. (2016).

Além dos padrões de segurança da informação, também existem os dispositivos legais, ou seja, a legislação para apoiar a área. Os dispositivos legais possuem o objetivo de padronizar a área da segurança da informação a sua legislação local, garantindo a proteção das informações (CISCO, 2018).

O Gabinete de Segurança Institucional da Presidência da República do Brasil divulgou em seu Departamento de Segurança da Informação (2019) os principais dispositivos legais de caráter federal, que podem ser visualizados no Apêndice A deste trabalho.

Outro recurso que a área de segurança da informação possui é a realização de testes para encontrar vulnerabilidades, que nada mais são do que falhas capazes de proporcionar riscos para a informação. A exploração de vulnerabilidades, que pode significar uma apropriação indevida de ativos de informação, ou mesmo a interrupção de operações on-line, pode resultar em impactos econômicos significativos para uma organização (ISACA, 2016; STEINBART *et al*., 2018).

Segundo Weishäupl, Yasasin e Schryen (2018) e Srinivas, Das e Kumar (2018), estima-se que, em 2015, crimes cibernéticos tenham causado um prejuízo/dano de US $ 315 bilhões (trezentos e quinze bilhões de dólares) em todo o mundo. Esse número fez com que a área da segurança da informação recebesse mais atenção de comunidades científicas, que, por sua vez, buscam formas de resolver os problemas de vulnerabilidades em sistemas de informação.

Os prejuízos causados por crimes cibernéticos explicam por que a descoberta de vulnerabilidades se tornou uma atividade extremamente difundida, e por que novas ameaças estão sendo constantemente analisadas e estudadas (DOBROVOLJC; TRČEK; LIKAR, 2017).

O volume e a variedade de vulnerabilidades existentes fazem com que a tomada de decisão para identificá-las seja rápida, diminuindo o risco de a informação ser violada. Com os dispositivos se tornando cada vez mais vulneráveis, aguardar um determinado tempo para aplicar uma correção pode aumentar o risco (CISCO, 2018).

Nagpure e Kurkure (2017), Bellegrade, Orvis e Helba (2010) e Zhao e Dai (2012) apresentam, em seus estudos, as vulnerabilidades mais comumente vistas em sistemas de informação. Tais vulnerabilidades são descritas na Tabela 2.

Tabela 2 – Vulnerabilidades mais comuns em sistemas de informação

Vulnerabilidades	Descrição
Estouro de Memória	Ocorre quando o usuário fornece dados que se estendem por um limite de tamanho da variável, fazendo com que as informações necessitem ser alocadas em outro local de memória, o que muitas vezes o software não consegue antecipar.
Negação de Serviço	É realizado quando o invasor fornece mais informações que o software pode acomodar. Em sistemas interconectados, faz com que serviços parem de funcionar, proporcionando erros em acessos.
Injeção SQL	Ocorre com a inserção de códigos maliciosos em uma requisição SQL Server.
Cross-Site Scripting	Ocorre quando uma aplicação web reúne conteúdo malicioso, onde o usuário é redirecionado para esta página por meio de hiperlinks, e-mails, etc.
Engenharia Social	Trata-se do acesso a informações não autorizadas aproveitando-se de outros usuários. Ou seja, utiliza-se da manipulação de usuários para conseguir acesso a informações ou para chegar a um determinado sistema.
Furto de Identidade	Ato pelo qual uma pessoa tenta se passar por outra, atribuindo-se uma falsa identidade.
Furto de Sessão	Ocorre quando um invasor obtém acesso a uma sessão de um usuário específico dentro do sistema.
Escalonamento de Privilégios	Significa que um usuário recebe privilégios de outros usuários. Esses privilégios podem ser usados para excluir os arquivos, visualizar informações particulares ou instalar programas indesejados, como vírus.
Expiração insuficiente da sessão	Consiste em uma consequência de uma sessão mal implementada. Devido a essa limitação, pode-se levantar níveis de design e implementação para obter acesso não autorizado para um aplicativo específico.
Fixação de Sessão	Ocorre quando se cria uma sessão válida em um servidor web por meio de uma sessão sequestrada anteriormente.
Passagem de Diretório	Ocorre quando se obtêm acesso a diretórios e arquivos restritos através de um usuário sem privilégios. Isso ocorre devido à a filtragem insuficiente na entrada de navegadores e perfis.
Clickjacking	Ocorre quando se insere um código malicioso em formulários HTML presentes em uma página web, como links e botões.
Exploração do Cache do Navegador	Ocorre quando se utiliza os arquivos do navegador que estão salvos em cache. Geralmente, esses arquivos costumam ter informações que o usuário solicita para o navegador “lembrar”.
Ataques de Repetição em Navegadores	As páginas web geralmente geram cookies de sessão exclusivo para cada sessão válida. Esses cookies contêm dados confidenciais como usuário e senha. Essa vulnerabilidade ocorre quando se pega os cookies de usuário e utiliza-os em outra máquina, fazendo uso das informações.

Fonte: Adaptado de Nagpure e Kurkure (2017), Bellegrade, Orvis e Helba (2010) e Zhao e Dai (2012)

Algumas vulnerabilidades são tão complexas que suas detecções são difíceis de prever. Um exemplo disso são as vulnerabilidades provenientes de erros em softwares, que podem ser geradas em diversos momentos, desde a definição dos requisitos do software até a sua codificação e seu desenvolvimento (SRIVASTAVA; KUMAR, 2017).

Segundo Srivastava e Kumar (2017), as vulnerabilidades mais complexas fazem com que administradores de sistemas de informação e profissionais e pesquisadores da área de segurança da informação busquem por novas abordagens para identificá-las e combatê-las.

Em seu estudo, Guo *et al*. (2018) apresentam uma forma de identificar a estrutura de dados explorados em vulnerabilidades de estouro de memória. Os autores abordam a análise e manipulação dessas vulnerabilidades já documentadas por meio da conversão de *strings* para seu valor numérico em ASCII.

As vulnerabilidades em sistemas de informação podem fazer com que as organizações aumentem seus custos consideravelmente, seja contratando empresas terceirizadas para prover segurança da informação ou mesmo arcando com as consequências de uma vulnerabilidade explorada (GARG; SIKKA; AWASTHI, 2018; FENG *et al*., 2019).

Segundo Nagpure e Kurkure (2017), as vulnerabilidades podem ser descobertas e analisadas de duas formas: pelo método de avaliação de vulnerabilidades e pelo processo de teste de intrusão. Enquanto o primeiro analisa ativos e sistemas de informação com o objetivo de encontrar vulnerabilidades sem explorá-las, o segundo analisa ativos e sistemas de informação para encontrar vulnerabilidades e, em seguida, explorá-las, a fim de se buscar novas informações ou outras vulnerabilidades.

2.2 TESTES DE INTRUSÃO

Teste de intrusão ou Pentest (*Penetration Testing*) é um tipo de teste de segurança da informação, que envolve a simulação de ataques de hackers para avaliar a segurança de sistemas de informação. Sua execução é feita geralmente de forma manual, seu ciclo de testes é longo quando comparado ao automático, e seu resultado é um relatório que enumera as vulnerabilidades encontradas e exploradas (ALMUBAIRIK; WILLS, 2016; TETSKYI; KHARCHENKO; UZUN, 2018).

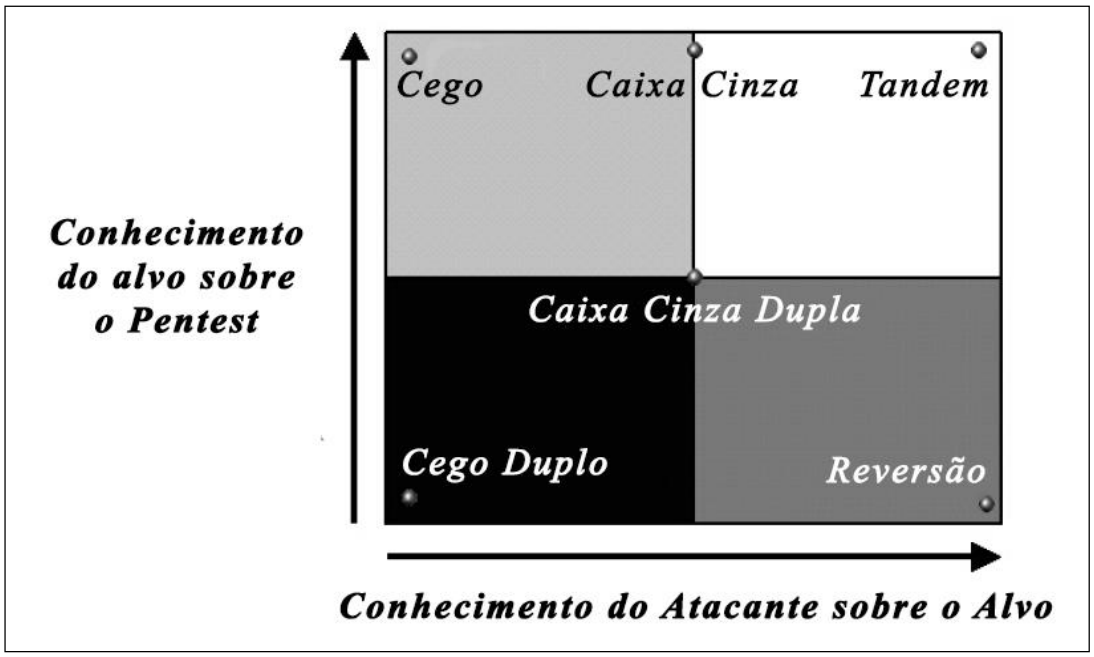
Os ataques de hackers simulados em um Pentest por profissionais de segurança da informação são tentativas de destruir, expor, alterar, desativar, roubar ou obter informações não autorizadas para, assim, avaliar a segurança de uma determinada informação (NAVARRO; DERUYVER; PARREND, 2018; ISO/IEC, 2016).

O Pentest surgiu no início dos anos 1970, quando o Departamento de Defesa dos Estados Unidos (*Department of Defense – DoD*) utilizou pela primeira vez uma técnica para demonstrar as vulnerabilidades em sistemas de informação. A partir disso, iniciou-se um projeto para desenvolver softwares, visando mitigar a exploração de vulnerabilidades para tornar os sistemas mais seguros (ALHASSAN *et al*., 2018).

A eficiência do Pentest faz com que sua realização seja parte essencial da estratégia de avaliar os possíveis riscos de uma organização. Sua utilização, para testar a segurança de um ou mais sistemas, envolve a busca por fraquezas em redes de computadores, revisão de código, erros de configuração de hardware e software, arquivos e dispositivos desprotegidos, dentre outros (ALMUBAIRIK; WILLS, 2016).

Segundo Herzog (2010), a metodologia *The Open Source Security Testing Methodology Manual* 3 (OSSTMM 3) define que o *Pentest* pode ser classificado com base na quantidade de informação que o profissional de segurança da informação sabe sobre seu alvo, e a quantidade de informação que o alvo sabe sobre o Pentest. Os tipos de Pentest definidos pela metodologia OSSTMM são apresentados na Figura 1.

Figura 1 - Tipos de *Pentest* definidos pela metodologia OSSTMM 3



Fonte: Traduzido de Herzog (2010).

A seguir, são descritos os tipos de Pentest definidos pela metodologia OSSTMM 3 e apresentados na Figura 1 (HERZOG, 2010).

- **Cego**: Também chamado de “Jogos de Guerra*”* ou “Interpretação de Papéis*”*. Neste tipo de Pentest, o profissional de segurança da informação que irá executá-lo não sabe nada sobre seu alvo, enquanto o alvo sabe que acontecerá o Pentest e que testes serão realizados.

- **Cego Duplo**: Também chamado de “Caixa Preta”. Neste tipo de Pentest, o profissional de segurança da informação não conhece nada sobre seu alvo, e o alvo não sabe que será atacado, tampouco quais testes serão feitos. É o tipo Pentest mais realista possível, aproximando-se de um ataque hacker, pois ambas as partes não sabem com o que irão lidar.

- **Caixa Cinza**: Também chamado de “Teste de Vulnerabilidade Puro”. Neste tipo de Pentest, o profissional de segurança da informação tem conhecimento parcial do alvo, e o alvo sabe que será atacado e também sabe quais testes serão realizados. Aqui, é possível simular o ataque dentro de um ambiente monitorado e controlado.

- **Caixa Cinza Dupla**: Também chamado de “Caixa Branca”. Neste tipo de Pentest, o profissional de segurança da informação tem conhecimento parcial do alvo, e o alvo sabe que será atacado, porém, não sabe quais testes serão executados. Esse teste é utilizado para simular um ataque vindo de dentro do sistema, isto é, vindo de um funcionário com acesso privilegiado, por exemplo.

- **Tandem**: Também chamado de “Caixa Cristal”. Neste tipo de Pentest, o profissional de segurança da informação tem conhecimento total sobre o seu alvo, e o alvo sabe que será atacado e também quais testes serão realizados. Esse tipo de Pentest se aproxima de um processo de auditoria, pois todos estão preparados e sabem o que vai ser realizado.

- **Reversão**: Também chamado de “Exercícios para Times Vermelhos”. Nesse tipo de Pentest, o profissional de segurança da informação possui conhecimento total do alvo, porém o alvo não sabe que será atacado, tampouco sabe quais testes serão executados. Esse tipo de Pentest é utilizado para testar o desempenho do time de resposta a incidentes.

De acordo com Bertoglio e Zorzo (2017), para auxiliar a execução do Pentest, existem cinco metodologias que abordam práticas voltadas para testes de segurança da informação. As metodologias são descritas na Tabela 3.

Tabela 3 – Metodologias que abordam testes de segurança da informação

Metodologia	OSSTMM	ISSAF	NIST SP 800-115	OWASP	PTES
Entidade Responsável	ISECOM	OISSG	NIST	OWASP	-
Última Atualização	2010	2006	2008	2014	2012
Específica para *Pentest*	Não	Não	Não	Não	Sim

Fonte: Adaptado de Bertoglio e Zorzo (2017).

Na Tabela 3, constam as cinco metodologias que abordam os testes de segurança das informações, juntamente com a entidade responsável por sua criação, sua última atualização, ou seja, o último ano de publicação ou atualização da metodologia, e também a sua especificidade para Pentest.

Das cinco metodologias apresentadas por Bertoglio e Zorzo (2017), apenas a PTES é específica para Pentest, enquanto as outras são mais abrangentes e tratam de outros tipos de testes de segurança da informação. Por ser específica para Pentest, a metodologia PTES tem recursos que as outras não possuem, como sugestão de ferramentas.

A seguir, são descritas as cinco metodologias:

a) O **Guia de Metodologia de Teste de Segurança de Código Aberto**, ou *The Open Source Security Testing Methodology Manual* (OSSTMM), é uma metodologia utilizada para gerenciar e realizar avaliações de segurança da informação de diversos níveis, processos de auditoria, Pentest e análise de códigos maliciosos estáticos e dinâmicos. Sua entidade responsável é o Instituto para Segurança e Metodologias Abertas, ou *The Institute for Security and Open Methodologies* (ISECOM), uma organização voltada para o desenvolvimento de metodologias para a área de segurança da informação (HERZOG, 2010).

b) O **Framework de Avaliação de Segurança de Sistemas de Informação**, ou *Information Systems Security Assessment Framework* (ISSAF), é uma metodologia que possui o objetivo de integrar ferramentas de gestão, controle e testes de segurança, realizar avaliações de políticas de segurança e buscar conformidades para a estrutura de TI. Sua entidade responsável é o Grupo de Serviços de Segurança de Informações Abertas, ou *Open Information Security Service Group* (OISSG), grupo responsável por desenvolver ferramentas e metodologias para a área de segurança da informação com foco na gestão de riscos (ISSAF, 2006).

c) O **Instituto Nacional de Padrões e Tecnologia**, ou *National Institute of Standards and Technology* (NIST**)** publicou um guia chamado de **NIST SP 800-115** - *Technical Guide to Information Security Testing and Assessment.* Esse guia traz uma metodologia que aborda os principais aspectos técnicos e procedimentos sobre a realização de testes de segurança de informação e auditorias (SCARFONE *et al*., 2008).

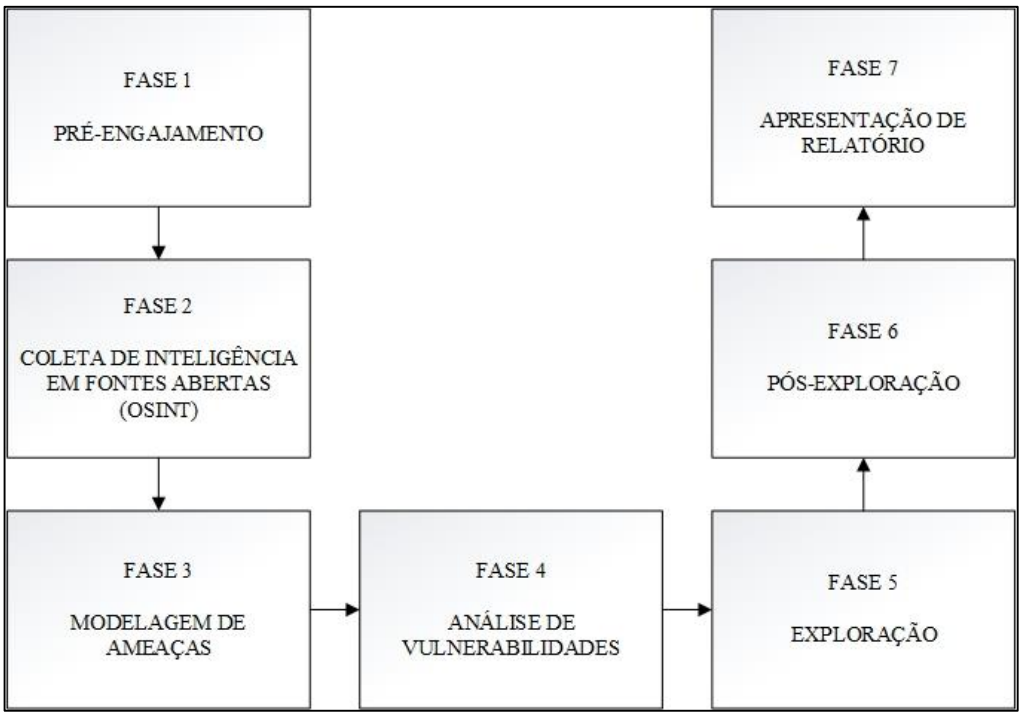
d) O **Guia de Testes OWASP**, ou *OWASP Testing Guide*, é uma metodologia mantida pela comunidade de profissionais de segurança da informação da entidade OWASP (*The Open Web Application Security Project*). Essa metodologia descreve quais são as melhores práticas, e técnicas e ferramentas necessárias para a execução de testes em páginas web (MEUCCI; MULLER, 2014).

e) A **Padronização de Execução de Testes de Intrusão**, ou PTES *Technical Guideline* (*The Penetration Testing Execution Standard*), é uma metodologia que abrange diversos itens relacionados ao planejamento e à execução de um Pentest, desde a comunicação inicial, para o desenvolvimento do seu escopo, até a realização de relatórios que capturam todo o processo (PTES TECHNICAL GUIDELINE, 2014).

Das cinco metodologias apresentadas, apenas a PTES não possui uma entidade responsável por se tratar de um recurso disponível em fontes abertas (*Open-Source*) desenvolvido por profissionais de segurança da informação. Além disso, a PTES *Technical Guideline* é a única metodologia específica para Pentest, diferente das demais que abordam testes de segurança da informação de uma forma mais abrangente ou mais focada, como é o caso da OWASP que só realiza testes em páginas web.

Quanto à execução de um Pentest, a metodologia PTES *Technical Guideline* descreve sete fases (PTES Technical Guideline, 2014), que podem ser observadas na Figura 2.

Figura 2 – As sete fases de um *Pentest* segundo a metodologia PTES Technical Guideline



Fonte: Adaptado de PTES Technical Guideline (2014).

A seguir, são descritas as sete fases do Pentest, segundo a metodologia PTES (PTES TECHNICAL GUIDELINE, 2014).

a) Fase 1 (Pré-engajamento): Nesta fase, define-se o escopo do Pentest e explica-se todos os procedimentos para o cliente ou responsável pelo sistema de informação em que o Pentest será realizado. Aqui, são definidos itens como: tipo de Pentest que será adotado, datas e horários, se será fornecida previamente alguma documentação sobre o sistema de informação, quais são os limites desse sistema de informação, ou seja, quais diretórios ou equipamentos não devem ser testados, por exemplo, as ferramentas que serão utilizadas, a metodologia e onde os resultados do Pentest ficarão armazenados.

b) Fase 2 (Coleta de Inteligência ou Inteligência de Fontes Abertas), do inglês, *Open Source Intelligence* (OSINT): Nesta fase, inicia-se o Pentest, ou seja, o levantamento de informações que apoiam a execução do Pentest. Aqui, buscam-se informações em páginas web, televisão, jornais ou qualquer outra fonte aberta que possa auxiliar na descoberta de informações sobre os sistemas de informações em que será realizado o Pentest. Tais informações podem ser: endereços de e-mail, estatísticas dos servidores on-line, informações sobre o domínio (como proprietário e faixas de endereços IP), e até mesmo algumas vulnerabilidades como dispositivos on-line desprotegidos e diretórios expostos.

c) Fase 3 (Modelagem de Ameaças): Nesta fase, define-se um modelo para representar todas as possíveis ameaças que podem ser exploradas no sistema de informação em que está sendo realizado o Pentest. Aqui, definem-se quais ameaças podem ser concretizadas e suas consequências para o proprietário do sistema de informação. Com esse modelo pronto, pode-se desenvolver outros para analisar a perspectiva de quem está executando o Pentest, do proprietário da organização onde está o sistema e do time de respostas a incidentes de segurança da informação.

d) Fase 4 (Análise de Vulnerabilidades): Nesta fase, efetua-se o mapeamento do sistema de informação em busca de vulnerabilidades, com base nas informações obtidas na fase 2. As vulnerabilidades podem variar desde uma configuração incorreta em um host até um serviço on-line desatualizado. As informações obtidas nesta fase podem ser: portas de rede abertas ou filtradas, nome de servidores e hosts, mecanismos de defesa on-line na rede, como firewalls e honeypots, arquivos e diretórios desprotegidos. Além disso, aqui é possível descobrir até se já existe algum código malicioso na rede.

e) Fase 5 (Exploração): Nesta fase, efetua-se a exploração das vulnerabilidades descobertas na fase anterior e na fase 2. Se as fases 2 e 4 tiverem sido concluídas corretamente, a execução da fase 5 será rápida e precisa; caso contrário, serão necessários novos ajustes e execuções nas fases 2 e 4, isto porque é preciso identificar com clareza a vulnerabilidade que será explorada aqui. Esta fase tem como objetivo conseguir ultrapassar as barreiras de segurança existentes, obter acesso a informações confidenciais e arquivos de configurações de serviços, acessar dispositivos remotamente sem a necessidade de autenticação por meio de um login e modificar ou excluir registros de banco de dados.

f) Fase 6 (Pós-exploração ou Bypass): Nesta fase, analisa-se o que foi possível extrair de um dispositivo comprometido, buscam-se meios para mantê-lo e verificam-se quais vantagens esse dispositivo pode trazer posteriormente. Por muitas vezes, pode-se tratar de uma máquina que possui arquivos sensíveis, ou seja, com informações confidenciais ou secretas, além de privilégios administrativos, e que é capaz de comprometer ainda mais a rede. Aqui, também acontece a criação de “backdoors”, ou seja, de maneiras alternativas de voltar a ter acesso às máquinas comprometidas. Por fim, nesta etapa, limpam-se todos os rastros deixados nas fases anteriores, removendo *logs* de acessos, arquivos utilizados e registros dos servidores de rede.

g) Fase 7 (Apresentação de Relatório): Nesta fase, desenvolve-se um relatório com as informações utilizadas no Pentest, além de todas as vulnerabilidades encontradas. As vulnerabilidades são enumeradas conforme suas características identificadas no Pentest, tais como Gravidade, Urgência, Impacto, dentre outras. As demais informações descobertas no Pentest podem se tratar de: arquivos coletados em diretórios desprotegidos, informações sensíveis descobertas como logins e redes privadas, anormalidades que ocorreram durante o Pentest, possíveis soluções para as vulnerabilidades descobertas e anomalias ou erros que surgiram com a execução do Pentest.

Além de poder optar pelo tipo de execução e metodologia do Pentest, também é possível escolher se a sua completa ou parcial execução será automática ou manual. Quanto ao Pentest parcialmente automático, executam-se apenas algumas fases ou tarefas utilizando softwares específicos, diferentemente da execução completa automática, em que todas as fases são realizadas com o auxílio de softwares (HATFIELD, 2018).

Segundo Chu e Lisitsa (2018), Nagpure e Kurkure (2017) e Hatfield (2018), o Pentest manual utiliza poucos ou nenhum software. Isto faz com que sua execução seja mais lenta do que a automática, que utiliza softwares em todas as fases. Assim, o *Pentest* automático permite que o profissional de segurança da informação que o esteja executando se concentre mais na análise dos resultados do que em sua execução.

Os softwares e algoritmos utilizados para executar o Pentest automático usam o conhecimento técnico disponível em sua composição, como as vulnerabilidades já encontradas e documentadas. Assim, uma atualização dos algoritmos e softwares se faz necessária sempre que novas vulnerabilidades forem surgindo (CHU; LISITSA, 2018).

Por causa das características citadas acima, o Pentest manual é mais utilizado em tarefas de Engenharia Social e Revisão de Código, enquanto o Pentest automático é utilizado em quase todas as suas fases (NAGPURE; KURKURE, 2017).

Vale destacar que a execução do Pentest inicia com a coleta de informações em fontes abertas (como pôde ser observado na fase 2 da Figura 2). Em tal fase, descobrem-se vulnerabilidades em sistemas, ou ativos de informação, e até mesmo informações que possam levar a vulnerabilidades. Essa fase do Pentest é chamada de Inteligência de Fontes Abertas ou OSINT (*Open Source Intelligence*) (KNOWLES; BARON; MCGARR, 2016; KOTHIA; SWAR; JAAFAR, 2019).

2.3 INTELIGÊNCIA DE FONTES ABERTAS

Inteligência de Fontes Abertas ou *Open Source Intelligence* (OSINT) é um conceito que aborda a busca, a coleta, o processamento, a análise e o uso de informações localizadas em fontes abertas, bem como as técnicas e ferramentas utilizadas (KOOPS; HOEPMAN; LEENES, 2013; HOWELLS; ERTUGAN, 2017).

Dois termos importantes relacionados ao OSINT são definidos pelo Departamento do Exército dos Estados Unidos da América (2012): Fontes Abertas e Informações Públicas Disponíveis.

As **Fontes Abertas** são definidas como qualquer pessoa, grupo, organização ou sistema que fornece informações sem expectativa de privacidade. As informações contidas nessas fontes não são protegidas contra divulgação pública. Embora estejam disponíveis ao público, as informações não devem ser abertas necessariamente.

Já as **Informações Públicas Disponíveis** são dados, fatos, instruções ou outro material publicado, compartilhado ou transmitido para consumo público em geral, sendo legalmente observado por qualquer indivíduo.

OSINT não se trata de um conceito novo, e surgiu como uma forma de solucionar problemas envolvendo negócios, militares, políticos e organizações que realizam serviços de inteligência, como *Central Intelligence Agency* (CIA), *Federal Bureau of Investigation* (FBI) e EUROPOL. Com o tempo, surgiram conferências sobre OSINT, fazendo com que as organizações e universidades participantes apresentassem pesquisas e técnicas sobre o tema. (HAYES; CAPPA, 2018; GLASSMAN; KANG, 2012).

A prática de coleta de informações não é recente, pois vem sendo discutida desde 1941, quando foi desenvolvido o *Foreign Broadcast Monitoring Service*, uma organização que tinha o objetivo de monitorar as transmissões de rádio entre Alemanha e Japão. Mais tarde, essa organização transformou-se em Centro de Fontes Abertas (CLARKE, 2015).

Desde a criação do Centro de Fontes Abertas até os dias atuais, inúmeras ferramentas e técnicas para a coleta de informação de fontes abertas foram desenvolvidas, como por exemplo o Google, o Maltego, o theHarvester e o Carrot2 (LEE; SHON, 2016).

Embora o *Foreign Broadcast Monitoring Service* tenha surgido em 1941, o OSINT foi definido apenas em 2001 com a publicação *The Open Source Intelligence Handbook* da Organização do Tratado do Atlântico Norte (OTAN, 2001). A publicação passou a definir OSINT como informações não confidenciais que foram deliberadamente descobertas, discriminadas, destiladas e disseminadas para um público-alvo a fim de abordar uma questão específica.

Em sua origem, OSINT era associado apenas a serviços de inteligência em organizações governamentais. Quem executava o OSINT ficava responsável pela análise das informações encontradas, que geralmente estavam ocultas ou escondidas. Com o passar do tempo, diante do crescente volume de informações e com a internet em ascensão, o OSINT começou a ser relacionado à busca de informações pertinentes, ou seja, que tenham algum conhecimento (MACIOŁEK; DOBROWOLSKI, 2013).

Portanto, a exploração de conteúdo da internet é uma ação indispensável na execução de OSINT. Grande parte desse conteúdo é pesquisável utilizando mecanismos de busca ou outras ferramentas. Com a Web 2.0, além de mídias sociais como Twitter e LinkedIn, há também outras plataformas que armazenam informações. Esse aumento de flexibilidade explica a crescente necessidade de segurança e proteção para as informações disponíveis na internet (KROMBHOLZ *et al*., 2015).

OSINT pode ser utilizado em uma ampla variedade de fontes abertas, como mídias sociais, blogues, páginas web governamentais que disponibilizam relatórios, páginas de geolocalizações, redes sociais, fóruns e comunidades, páginas que exibem imagens de satélite, publicações acadêmicas, banco de dados de vulnerabilidades, além de várias outras fontes disponíveis na internet e outros recursos de mídia (QUICK; CHOO, 2018; PELLET, SHIAELES; STAVROU, 2019; SETTANNI *et al*., 2017).

Quanto às informações que podem ser encontradas em fontes abertas, Rico *et al*. (2018) apresentam em seu estudo alguns exemplos, tais como: endereços IP e domínios, e-mails, documentos pessoais de identificação como RG e CPF, fotos, números de telefone, situação socioeconômica, documentos de afiliação a serviços como planos de saúde, antecedentes criminais, informações contratuais de trabalho, dentre outros.

Pela riqueza, variedade e importância da informação descoberta por OSINT, as organizações começam a reconhecê-la como um ponto crucial da atividade de monitoramento

e testes. Tal informação pode ser descoberta ou obtida legalmente por meio de compra, solicitação, coleta ou observação, além de ser classificada como não-confidencial (WATTERS; LAYTON, 2016).

Pesquisadores da área de tecnologia da informação direcionaram os estudos sobre OSINT para um caminho diferente do tradicional voltado para fins militares. O novo objetivo é que o OSINT seja executado de forma automática e utilize técnicas de Inteligência Artificial (IA) e de Processamento de Linguagem Natural (PLN) para minerar, estruturar, traduzir e extrair conhecimento das informações descobertas (NOUBOURS; PRITZKAU; SCHADE, 2014).

Assim, a aplicação de OSINT na área de tecnologia da informação está direcionada em três principais áreas: Segurança da Informação, Marketing Digital e Inteligência Artificial (GALINDO *et al.*, 2019).

Para a área da segurança da informação, OSINT é utilizado em Pentest para buscar vulnerabilidades ou informações que ajudem a revelar vulnerabilidades, sejam elas físicas, digitais, eletrônicas ou humanas (PTES Technical Guidelines, 2014). No marketing digital, é utilizado para mineração de opinião em mídias sociais.

Segundo Galindo *et al*. (2019), para a área da IA, OSINT é utilizado em conjunto com algoritmos de aprendizagem de máquina ou PLN para automatizar sua execução e extrair conhecimento dos resultados encontrados.

Em Pentest, as informações descobertas com a utilização de OSINT podem ser: malwares (códigos maliciosos), *phishings*, dispositivos infectados on-line, *logs* de servidores, diretórios abertos, credenciais de usuários, manuais de dispositivos on-line como impressoras 3D e câmeras, sistemas de defesa como honeypots e firewalls, arquivos não indexados, dentre outros (GONG; CHO; LEE, 2018; VACAS; MEDEIROS; NEVES, 2018; MILLER *et al*., 2018).

2.3.1 Abordagens de OSINT para Pentest

As abordagens de OSINT para Pentest são formas utilizadas para executá-lo de maneira automática com o objetivo de melhorar seu desempenho. A principal vantagem em se utilizar uma abordagem de OSINT em um Pentest é reduzir a execução de testes repetitivos com o auxílio de softwares ou ferramentas (POUCHARD, DOBSON e TRIEN, 2009).

Na revisão sistemática da literatura realizada que pode ser encontrada no capítulo 3, seção 3.4 deste trabalho, encontrou-se um total de três abordagens de OSINT para *Pentest* e uma estrutura básica de uma abordagem OSINT para *Pentest*. Apresenta-se na tabela 4 as três abordagens de OSINT para *Pentest* identificadas neste trabalho.

Tabela 4 – Três abordagens de OSINT para Pentest identificadas neste trabalho

Abordagem	Fases	Autores	Ano
Abordagem OSINT para Apoiar Operações de Segurança Cibernética	4 Fases	Ricardo Andrés Pinto Rico; Martin José Hernández Medina; Cristian Camilo Pinzón Hernández; Daniel Orlando Díaz López; Juan Carlos Camilo García Ruíz.	2018
Abordagem OSINT para Inspecionar Sistemas de Infraestruturas Críticas	4 Fases	Seokcheol Lee; Taeshik Shon.	2016
Abordagem OSINT para Obter Informações Sobre Inteligência de Ameaças Cibernéticas	3 Fases	Ke Li; Hui Wen; Hong Li; Hongsong Zhu; Limin Sun.	2018

Fonte: o Autor (2020).

Descrevem-se, a seguir, as três abordagens de OSINT para Pentest identificadas neste trabalho.

A primeira abordagem de OSINT para Pentest é proposta por Rico *et al*. (2018). Os autores apresentam uma abordagem de OSINT composta por quatro fases para apoiar operações de segurança cibernética, conforme pode-se observar na Figura 3.

Figura 3 – Abordagem OSINT para apoiar operações de segurança cibernética

Fonte: Adaptado de Rico *et al*. (2018).

As fases da abordagem OSINT proposta por Rico *et al*. (2018), para apoiar operações de segurança cibernética, ocorrem da seguinte maneira:

Na primeira fase, chamada de **Definição das Fontes Abertas**, definem-se quais fontes abertas serão selecionadas para buscar as informações, e quais são as informações necessárias para a execução da abordagem, como vulnerabilidades e dados confidenciais. As fontes abertas podem ser: mídias sociais, páginas web corporativas e blogues.

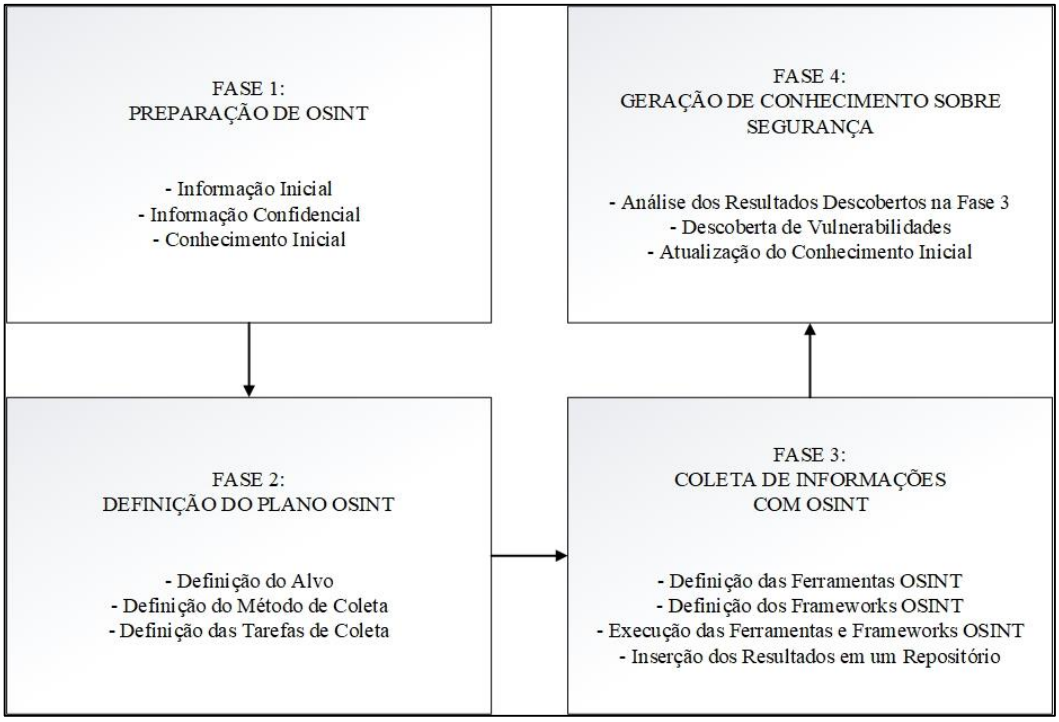
Na segunda fase, chamada de **Seleção e Execução de OSINT**, definem-se quais ferramentas OSINT, *frameworks* OSINT e mecanismos de busca, serão utilizados para, em seguida, executá-los nas fontes abertas definidas na primeira fase.

Na terceira fase, chamada de **Processamento dos Resultados**, acontece a análise dos resultados obtidos da execução das ferramentas OSINT, *frameworks* OSINT e mecanismos de busca na segunda fase. Os resultados são processados para que, desta forma, possa-se eliminar informações duplicadas e falsos positivos. Por fim, os dados obtidos dos resultados são estruturados de modo que seja possível executar uma análise para extrair conhecimento.

Na quarta fase, chamada de **Análise dos Resultados**, os resultados processados na fase anterior são analisados para se extrair conhecimento. Tal conhecimento pode significar uma previsão de possíveis ataques a uma determinada rede, encontrar padrões de ataque e relações entre as vulnerabilidades presentes em sistemas de informação, como páginas web.

A segunda abordagem identificada neste trabalho foi proposta por Lee e Shon (2016), que desenvolveram uma abordagem OSINT para inspecionar vulnerabilidades em sistemas de informação responsáveis por gerenciar infraestruturas críticas, como represas, metalúrgicas e hidrelétricas. A abordagem feita pelos autores é apresentada na Figura 4.

Figura 4 – Abordagem de OSINT para inspecionar sistemas de controle de infraestruturas críticas



Fonte: Adaptado de Lee e Shon (2016).

A abordagem proposta por Lee e Shon (2016) se inicia com a primeira fase chamada **Preparação de OSINT**. Nessa fase, verificam-se as informações iniciais que se possui sobre o sistema em que se irá executar a abordagem, sejam elas públicas ou confidenciais.

O conhecimento inicial pode ser qualquer informação que auxilie a execução da abordagem, como por exemplo: os endereços IP dos ativos de informação presentes no sistema, a linguagem de programação utilizada para o desenvolvimento do sistema e a identificação do responsável pela gestão do sistema.

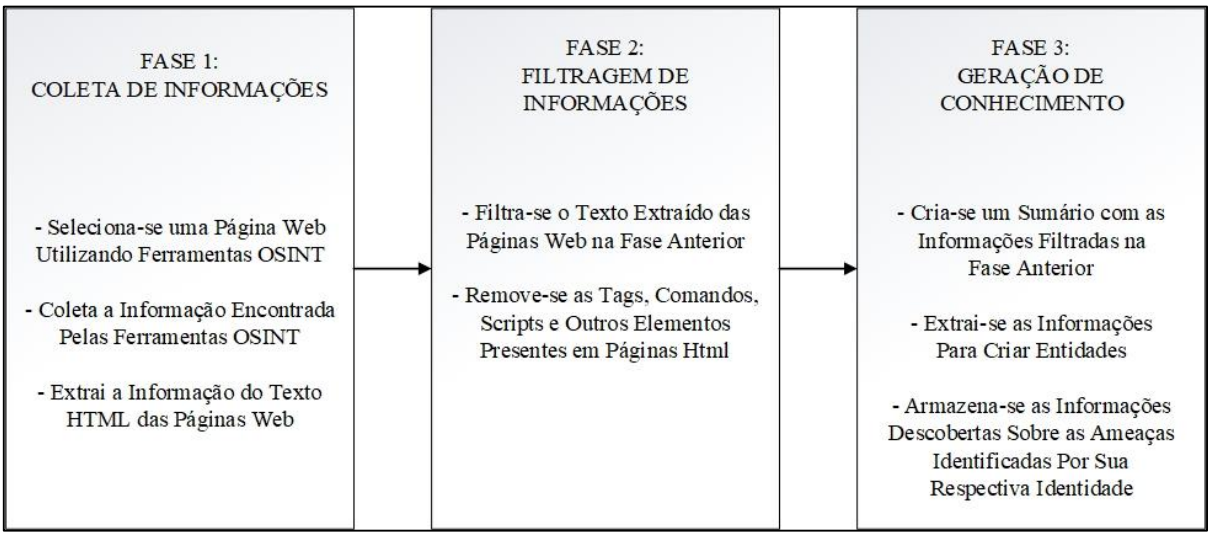
Na segunda fase, denominada **Definição do Plano OSINT**, prepara-se a execução de OSINT. Nessa fase, define-se o "alvo", ou seja, onde o OSINT será aplicado e quais tipos de informações devem ser coletados. Também são definidas quais tarefas serão executadas para encontrar as informações desejadas e qual método de coleta será utilizado.

Na terceira fase, chamada de **Coleta de Informações com OSINT**, definem-se as ferramentas e *frameworks* OSINT que serão executadas. Os resultados dessa execução são armazenados em um repositório para análise posterior.

Na quarta fase, denominada **Geração de Conhecimento sobre Segurança**, analisam-se os resultados descobertos na terceira fase, a fim de gerar um conhecimento sobre a segurança do sistema. Esse conhecimento pode representar a descoberta de novas vulnerabilidades, possibilitando o desenvolvimento de ações para corrigi-las. O conhecimento descoberto é utilizado para atualizar o conhecimento inicial utilizado na primeira fase da abordagem.

A terceira abordagem identificada neste trabalho foi proposta por Li *et al*. (2018), que descrevem a utilização de uma abordagem OSINT para obter informações sobre inteligência de ameaças cibernéticas, do inglês, *Cyber Threat Intelligence* (CTI). A abordagem feita pelos autores é apresentada na Figura 5.

Figura 5 – Abordagem OSINT para obter informações sobre CTI



Fonte: Adaptado de Li *et al*. (2018).

A abordagem desenvolvida por Li *et al*. (2018) se inicia com a primeira fase chamada **Coleta de Informações**. Nessa fase, seleciona-se uma página web que contenha informações sobre uma vulnerabilidade CTI para, em seguida, coletá-la. A coleta é feita extraindo-se o conteúdo da página web.

Na segunda fase, denominada **Filtragem de Informações**, filtra-se o texto extraído das páginas web identificadas na primeira fase. Para isto, limpa-se o texto, removendo o conteúdo que não possui relação com a vulnerabilidade CTI, como tags (comandos) em HTML, scripts, formulários, dentre outras informações presentes em uma página web.

Na terceira fase, denominada **Geração de Conhecimento**, realiza-se a análise dos textos filtrados na fase anterior. Os textos são classificados e sumarizados para que se possam criar as

entidades, que, por sua vez, funcionam como modelos para que seja possível classificar novas ameaças CTI que tenham características semelhantes com as entidades geradas.

As abordagens citadas anteriormente apresentam o desenvolvimento e a aplicação de abordagens OSINT em testes de segurança da informação, como o Pentest. Observando essa tendência, Zhao, Cao e Liu (2015) propuseram uma estrutura básica com o objetivo de padronizar o desenvolvimento de abordagens, ferramentas e *frameworks* OSINT.

A principal vantagem em utilizar uma estrutura básica para o desenvolvimento de abordagens, ferramentas ou *frameworks* OSINT é a possibilidade de se executar diversas técnicas de mineração de texto, ou mesmo técnicas de análise computacional, nos resultados encontrados para gerar conhecimento novo (ZHAO; CAO; LIU, 2015).

Ainda segundo Zhao, Cao e Liu (2015), a estrutura básica é dividida em quatro componentes principais: Ferramentas Para Aquisição dos Dados, Armazenamento dos Dados, Análise Computacional e Aplicação; porém, há um quinto componente denominado Sistema Operacional. A estrutura desenvolvida pelos autores é apresentada na Figura 6.

Figura 6 – Estrutura básica para o desenvolvimento de abordagens, ferramenta e *framework* OSINT

Aplicação	- Gerar Conhecimento - Apoiar Serviços de Inteligência - Apoiar Testes de Segurança da Informação;
Análise Computacional	- Associação - Agrupamento - Classificação - Indexação - Previsão.
Armazenamento dos Dados	- Banco de Dados Relacional - Banco de Dados Não-Relacional - Outros tipos de Repositórios.
Ferramentas Para Aquisição dos Dados	- Ferramentas OSINT - Frameworks OSINT - Mecanismos de Busca.
Sistema Operacional	- Sistema Operacional Linux - Sistema Operacional Windows - Sistema Operacional Android

Fonte: Adaptado de Zhao, Cao e Liu (2015).

Os tópicos apresentados na Figura 6 e representados pela cor cinza são descritos pelos autores como componentes principais, pois os resultados da execução de OSINT dependem da escolha correta desses componentes. Já o item Sistema Operacional, destacado em azul, é descrito pelos autores como um componente básico, pois não interfere nos resultados obtidos da execução de OSINT.

Para o desenvolvimento de abordagens, ferramentas e *frameworks* OSINT, Zhao, Cao e Liu (2015) sugerem o desenvolvimento de uma estrutura definida por:

- Qual sistema operacional será utilizado;
- Quais ferramentas, *frameworks* e abordagens OSINT serão utilizados;
- Quais bancos de dados serão utilizados;
- Qual análise computacional será utilizada; e
- Qual o objetivo da execução de OSINT.

Inicialmente, escolhe-se em qual sistema operacional será executado o OSINT: Windows, Linux ou mesmo Android; tal decisão depende do próximo passo que é a escolha das ferramentas, *frameworks* e abordagens OSINT para aquisição dos dados.

As ferramentas selecionadas para a execução de OSINT deve ser compatível com o sistema operacional escolhido anteriormente. Após definir a forma de aquisição dos dados, determinam-se quais bancos de dados serão utilizados, tanto para apoiar a execução de OSINT quanto para armazenar os resultados encontrados. Feito isto, definem-se quais análises computacionais serão realizadas nos dados encontrados, se os dados devem ser agrupados, classificados, associados ou indexados.

Por fim, utilizam-se as informações obtidas na análise computacional em um objetivo pré-determinado, que pode ser: a aplicação de um serviço dentro de uma empresa, a sintetização de conhecimento para apoiar agências governamentais, a visualização de informações sobre segurança da informação ou até o suporte à execução de testes de segurança da informação.

Quando se executa o OSINT em um determinado teste de segurança da informação como o Pentest, as informações descobertas podem ser utilizadas não somente para descobrir vulnerabilidades e dados que possam levar a vulnerabilidades, como também para desenvolver um plano de conscientização dos colaboradores, tornando-os capazes de identificar problemas

de segurança da informação, como por exemplo: funcionários que compartilham informações corporativas em redes sociais (KNOWLES; BARON; MCGARR, 2016).

2.4 GOOGLE HACKING

Pelo fato de o código das páginas web ser aberto e estar acessível pela internet, hackers aproveitam de sua estrutura para encontrar vulnerabilidades com uma maior facilidade do que em outros tipos de sistemas. É possivel determinar a versão do código e a estrutura utilizada por uma página web apenas pesquisando por *strings*, ou seja, sequência de caracteres específicos em mecanismos de busca (MANSFIELD-DEVINE, 2015).

Mecanismos de busca são ferramentas de recuperação de informações em que os usuários inserem palavras-chave para consulta e, subsequentemente, conseguem trazer resultados de forma automática. Alguns exemplos de mecanismos de busca são: *Google*, *Yahoo* e *Baidu.* (CARRIÓN; PUNTES; LUQUE, 2017; CHAO *et al*., 2016).

Segundo Cambazoglu (2007), um mecanismo de busca é normalmente composto por três componentes principais: um rastreador, também chamado de *crawler*, um indexador e um processador de *strings* de consultas, ou de sentenças de consulta.

A seguir, são descritos os principais componentes de um mecanismo de busca: *Crawler*, Indexador e Processador de *String* de Consulta.

O **Rastreador ou *Crawler*** é o responsável por localizar e armazenar o conteúdo que está na internet. Esse conteúdo é utilizado nas buscas realizadas pelos usuários, por isso, os rastreadores estão a todo tempo ativos procurando novos conteúdos na internet.

O **Indexador** é o componente que analisa e efetua o download do conteúdo encontrado pelo rastreador. Sua execução é simultânea a do rastreador, permitindo a criação de um índice compacto para ser consultado pelo mecanismo de busca.

Já o **Processador de *Strings* de Consulta** é o responsável por avaliar as consultas que os usuários realizam e retornar para eles as páginas relevantes.

Os mecanismos de busca são utilizados pelos usuários para procurar informações na internet. Quando necessitam buscar informações específicas, eles utilizam combinações de

palavras-chaves e operadores lógicos para criar uma *string* mais refinada, capaz de trazer resultados mais relevantes (DEULKAR; NARVEKAR, 2015).

Pesquisadores de diversas áreas vêm estudando os mecanismos de busca em diferentes abordagens, isto é, na busca de produtos, publicações científicas e nas áreas de marketing social, economia, política e segurança da informação (SCHIMIDT; SCHNITZER; RENSING, 2016).

Roy *et al*. (2017) apresentam uma prática para Pentest que utiliza o mecanismo de busca Google com o objetivo encontrar vulnerabilidades em páginas na internet apenas utilizando *strings* específicas, ou seja, uma determinada sequência de caracteres que podem ser compostos ou não por operadores avançados do Google.

A *string* utilizada no Google para procurar vulnerabilidades é denominada Dork, já a prática para Pentest que utiliza o Google e as Dorks é denominada Google Hacking ou Google Dorking (Roy *et al*., 2017). Na Tabela 5, são apresentados os cinco principais itens para a prática do Google Hacking.

Tabela 5 – Cinco principais itens para a prática do Google Hacking

Item	**Descrição**
Uso de Sistema de Cache	A prática do Google Hacking utiliza o sistema de cache do Google para ir diretamente para um *snapshot* de uma página web. Dessa forma, consegue-se extrair informações da página sem adentrar o domínio, conseguindo, assim, consultar as páginas sem estabelecer nenhuma conexão direta com o destino.
Uso de Dorks	As Dorks são *strings* que podem ser compostas por operadores de busca do Google. As Dorks são utilizadas na prática do Google Hacking para encontrar vulnerabilidades ou informações que ajudem a revelar vulnerabilidades em páginas web.
Descoberta de Recursos de Rede	Ao combinar operadores de busca, a prática do Google Hacking consegue obter listas de endereços de servidores e serviços disponíveis em um determinado domínio, além de poder encontrar páginas que estão conectadas a uma determinada URL.
Coleta de Arquivos	A prática do Google Hacking descobre não somente vulnerabilidades na estrutura de uma página web, como também arquivos que estão abertos ao público, como: *logs* de senhas (explícitas, hash, criptografadas, etc.), logins, banco de dados, dentre outros.
Google Hacking Database	O Google Hacking Database (GHDB) é um banco de dados com milhares de Dorks testadas e validadas pela *Offensive Security*.

Fonte: Adaptado de Roy *et al*. (2017)

Em Pentest, a prática do Google Hacking é utilizada na fase inicial, também chamada de “Reconhecimento” ou “OSINT”. Essa busca realizada pelo Google Hacking é classificada como “Reconhecimento Passivo”, pois coleta informações sobre redes, páginas ou sistemas on-line sem ser invasivo (FAN; LI; ZHANG, 2018).

McGuffee e Hanebutte (2013) e Kalech (2019) abordam alguns tipos de informações que podem ser encontradas com o Google Hacking. A saber: nome de servidores ativos no sistema, diretórios abertos, cópias de arquivos ilegais, ranges de endereços IPs, nome de servidores, informações críticas sobre sistemas SCADA, serviços e equipamentos que se conectam à internet, como câmeras, impressoras, aparelhos telefônicos e roteadores.

Mider, Garlicki e Jan (2019) classificam a execução do Google Hacking em 3 tipos principais: Google Hacking por OSINT branco, Google Hacking por OSINT cinza e Google Hacking por OSINT preto. Segundo os autores, as cores branco, cinza e preto fazem referência à ética de quem pratica o Google Hacking. O branco representa o profissional ético, o preto representa o hacker, e o cinza representa um indivíduo que, por muitas vezes, está presente em ambos os lados, branco e preto.

O Google Hacking por OSINT branco envolve tarefas como: busca por páginas web deletadas ou arquivadas, busca por informações de usuários, como e-mail, endereços, perfis de redes sociais e números de telefones, além de busca por arquivos que possuem informações sensíveis, tais como páginas similares, páginas com referências ou com históricos de modificações.

Para o Google Hacking por OSINT cinza, as tarefas envolvem o acesso a recursos que não estão visíveis para usuários fora de um domínio específico, mas que podem ser vistos pelo profissional que trabalha no domínio. Suas buscas por informações vão desde arquivos de diversas extensões, como doc, jpg, iso e pdf, até arquivos de configurações de serviços e servidores on-line, como IIS, SQL Database e diretórios Apache.

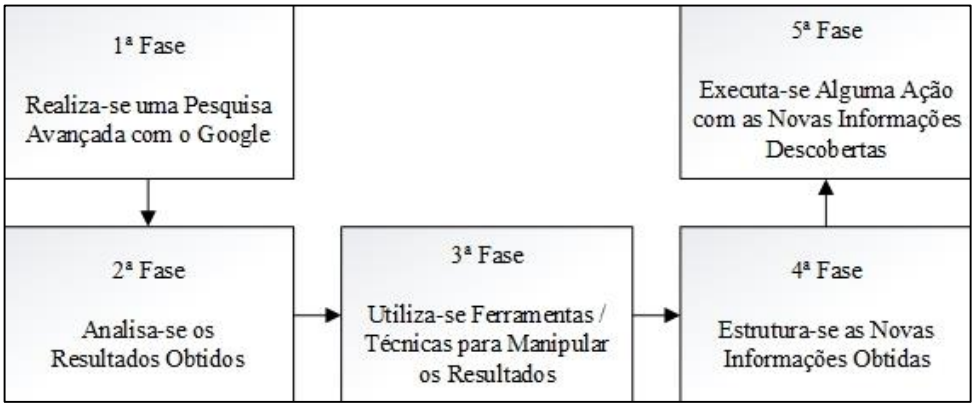
Por fim, a prática do Google Hacking por OSINT preto envolve atividades ilegais e antiéticas. Suas tarefas abrangem a obtenção de informações por meio de interceptação ou invasão e o acesso a dados pessoais como: logins, senhas e documentos de identificação, além de acesso à configuração de dispositivos pessoais, permitindo tomar o controle deles.

Para compreender o impacto que a prática do Google Hacking pode causar, Abdelhalim e Traore (2007) explicam em seu estudo a utilização do Google Hacking para encontrar indícios que podem ocasionar furtos de identidade e fraudes financeiras. Ao utilizar Dorks específicas para esse problema, compostas por palavras-chave como números de previdência social, data de nascimento e endereços residenciais, conseguiu-se encontrar informações sensíveis como: currículos on-line, extratos de empréstimos e petições públicas sobre dissolução de casamento.

A prática do Google Hacking em Pentest é classificada como "Reconhecimento Passivo", ou seja, não adentra as páginas web onde o Pentest está sendo executado, tornando-a difícil de ser detectada (MUNIR *et al*., 2015).

Um cenário explicando como a prática do Google Hacking funciona é apresentado no trabalho de Munir *et al*. (2015). Nele, os autores discutem como detectar, mitigar e calcular o risco de tentativas de intrusão classificadas como "Reconhecimento Passivo". Tal cenário é apresentado na Figura 7.

Figura 7 - Cenário sobre o funcionamento da prática do Google Hacking



Fonte: Adaptado de Munir *et al*., (2015)

Na Figura 7, demonstra-se um cenário sobre a prática do Google Hacking. Na primeira fase, realiza-se o Google Hacking com Dorks pré-definidas. Com as Dorks, verificam-se quais informações serão obtidas e quais serão utilizadas. Em seguida, na segunda fase, analisam-se os resultados obtidos, a fim de extrair alguma informação sobre vulnerabilidade.

Na terceira fase, utilizam-se ferramentas e técnicas para manipular os resultados obtidos com as Dorks para encontrar novas informações. Na quarta fase, estruturam-se as informações obtidas com a utilização das ferramentas e técnicas da fase anterior. Por fim, na quinta fase, utilizam-se as novas informações descobertas para descobrir novas vulnerabilidades.

Bae, Lim e Cho (2016) apresentam uma forma de executar o Google Hacking de maneira automática em páginas web governamentais. Para isso, eles utilizaram o software *SiteDigger3.* Com esse estudo, constatou-se que o software possui limitações na quantidade de Dorks, impossibilitando a inserção de novos parâmetros. Desta forma, a ferramenta trouxe poucos resultados sobre vulnerabilidades, portanto, os autores sugerem buscar outros meios para executar o Google Hacking de forma automática.

2.4.1 DORKS

As Dorks são "*strings*" que podem ser compostas por palavras e/ou parâmetros específicos desenvolvidos para mecanismos de busca com o objetivo de coletar informações sobre vulnerabilidades ou dados que auxiliem na sua busca. Na literatura, são descritas diversas Dorks para diferentes fins, ou seja, para encontrar sites vulneráveis, informações confidenciais ou arquivos expostos (SIMON, 2016; TOFFALINI *et al*., 2016).

Toffalini *et al*. (2016) apresentam outra definição para as Dorks, pois, para eles, as definições encontradas na literatura explicam as Dorks como "*strings*" compostas por parâmetros, embora nem todas usem parâmetros necessariamente. Para os autores, as Dorks são "*strings*" usadas em mecanismos de busca com o objetivo de encontrar informações específicas, com base na estrutura do site e não necessariamente em seu conteúdo.

Pan *et al*. (2012) descrevem categorias que podem ser utilizadas para classificar as palavras e os parâmetros que compõem as Dorks. Na Tabela 6, apresentam-se as categorias, juntamente com sua descrição e exemplos.

Tabela 6 – Categorias para classificar os componentes de uma Dork

Categoria	**Descrição**	**Exemplos**	**Utilização**
GRAM	Operadores Avançados do Google	Intext, Filetype, Intitle, Inurl, Site	Direciona a busca para a estrutura do Site
WEB	Vulnerabilidades em Tecnologias Web	Phpmyadmin, Wordpress	Direciona a busca para tecnologias Web
SCRIPT	Extensões de Páginas Web	Php, AspX, Asp, Jsp	Direciona a busca para tipos específicos de páginas web
DOC	Arquivos Desprotegidos	Doc, Pdf, Docx, Xls, Xlsx	Direciona a busca para tipos específicos de arquivos indexados em páginas web
BD	Arquivos de Banco de Dados	Sql, Mdb, Myd	Direciona a busca para tipos específicos de banco de dados indexados em páginas web

Fonte: Adaptado de Pan *et al*., (2012)

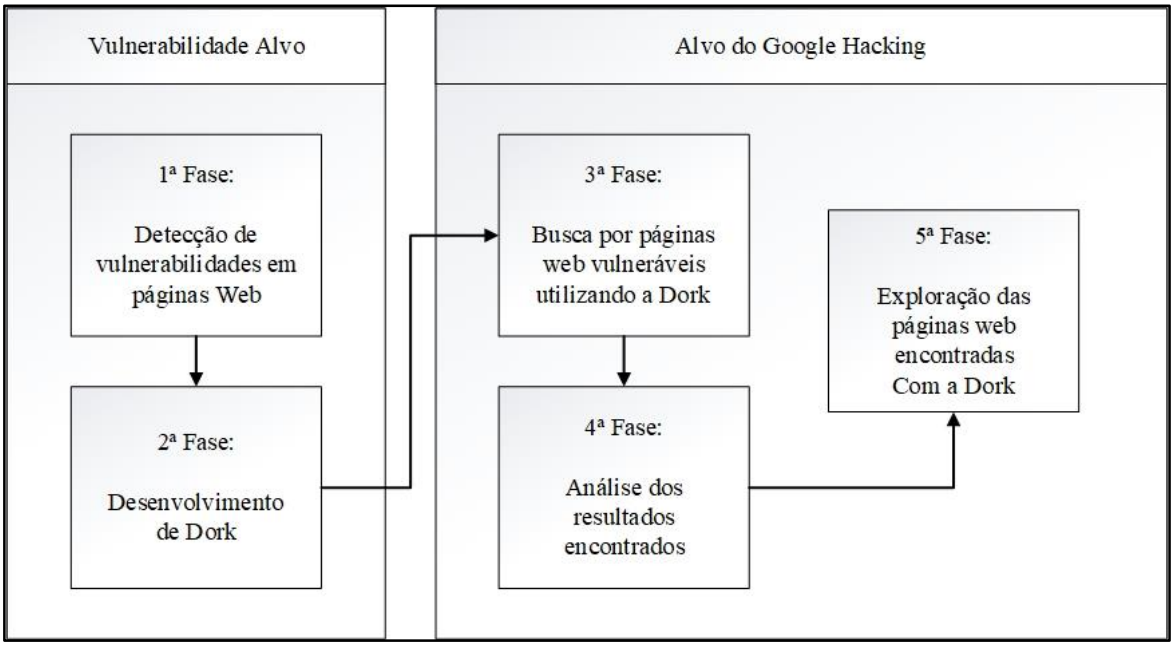
Exemplifica-se, com base na Tabela 6, uma Dork composta pelos seguintes elementos:

Inurl: “.gov.br” Intext: “Senhas.xlsx” OR “logins.doc”

Inurl e Intext pertencem à categoria GRAM, pois são operadores avançados do Google. São eles os responsáveis por direcionar a busca para a estrutura de um determinado site. O parâmetro Inurl procura por sites que contenham “.gov.br” em sua URL, enquanto o parâmetro Intext procura por sites que contenham em seu conteúdo arquivos da categoria DOC com o nome “Senhas.xlsx” ou “logins.doc”. Assim, a Dork pode ser utilizada para procurar por arquivos “Senhas.xlsx” ou logins.doc” em sites que contenham “.gov.br” em sua URL.

Em relação ao desenvolvimento e à utilização de Dorks, Zhang, Notani e Gu (2015) descrevem um processo geral para a prática do Google Hacking em Pentest, o que pode ser observado na Figura 8.

Figura 8 - Processo de desenvolvimento e utilização de Dorks



Fonte: Adaptado de Zhang, Notani e Gu (2015)

Na Figura 8, percebe-se a divisão do processo de desenvolvimento de Dorks em duas partes: Vulnerabilidade Alvo e Alvo do Google Hacking. A primeira parte envolve o desenvolvimento da Dork, cujo objetivo é detectar vulnerabilidades presentes em páginas web. Já a segunda parte envolve a utilização da Dork, que como por objetivo utilizar a Dork desenvolvida em outras páginas web para encontrar mais ocorrências da vulnerabilidade detectada na primeira parte.

A Vulnerabilidade Alvo é dividida em duas fases:

1ª Fase (Detecção de Vulnerabilidades em Páginas Web): Nesta fase, identifica-se uma vulnerabilidade em uma página web, que pode ser a versão desatualizada de um serviço de gerenciamento de banco de dados por exemplo.

2ª Fase (Desenvolvimento de Dork): Nesta fase, desenvolve-se uma Dork para procurar sites em potencial que possuam vulnerabilidade.

O Alvo do Google Hacking é dividido em três fases:

3ª Fase (Busca por Páginas Web Vulneráveis Utilizando a Dork): Nesta fase, efetua-se a busca por páginas web vulneráveis utilizando o Google e pesquisando com a Dork desenvolvida na segunda fase.

4ª Fase (Análise dos Resultados Encontrados): Nesta fase, verificam-se os resultados obtidos da busca realizada na terceira fase.

5ª Fase (Exploração das Páginas Web Encontradas com a Dork): Nesta fase, exploram-se as páginas web vulneráveis encontradas, verificando-se que tipo de informação é possivel extrair delas.

Segundo Meucci e Muller (2014), Zhang, Notani e Gu (2015) e Mider, Garlicki e Jan (2019), a maior concentração de Dorks validadas e documentadas no mundo está disponível no Google Hacking Database (GHDB). O GHDB é o maior e mais representativo banco de dados de Dorks on-line do mundo. Uma desvantagem da base é possuir poucos atributos: somente o texto da Dork, o autor que a publicou na base e categoria ao qual a Dork pertence.

As Dorks disponíveis no GHDB são classificadas em 14 categorias, baseando-se na sua funcionalidade, ou seja, no tipo de vulnerabilidade que elas buscam. As categorias são apresentadas na Tabela 7.

Tabela 7 – As catorze categorias de Dorks do Google Hacking Database

Categoria de Dork	**Descrição**
Foothold	Páginas que exibem algum rastro de vulnerabilidades
Files Containing Usernames	Páginas que contêm arquivos com nomes de usuários ou logins
Sensitive Directories	Páginas que possuem diretórios sensíveis desprotegidos
Web Server Detection	Páginas que possuem informações desprotegidas sobre seu servidor web

Categoria de Dork	Descrição
Vulnerable Files	Páginas que contêm arquivos desprotegidos
Vulnerable Servers	Páginas que contêm servidores desprotegidos
Error Messages	Páginas que revelam vulnerabilidades por meio de mensagens de erro
Files Containing Juicy Info	Páginas que contêm arquivos de configurações desprotegidos
Files Containing Passwords	Páginas que contêm arquivos com senhas desprotegidos
Sensitive Online Shopping Info	Páginas sobre e-commerce que exibem informações desprotegidas
Network of Vulnerability Data	Páginas que exibem dados vulneráveis sobre a estrutura de uma rede
Pages Containing Login Portals	Páginas que contêm portais de logins vulneráveis
Various Online Devices	Páginas que contêm dispositivos on-line desprotegidos
Advisories and Vulnerabilities	Páginas que contêm vulnerabilidades provenientes de anúncios

Fonte: Adaptado de Zhang, Notani e Gu (2015) e Meucci e Muller (2014)

Segundo Mider, Garlicki e Jan (2019), na prática do Google Hacking, a utilização das Dorks contidas no GHDB permite encontrar vulnerabilidades em páginas web já na fase inicial de um Pentest, chamada de “Reconhecimento” ou OSINT.

2.5 INTELIGÊNCIA ARTIFICIAL

Inteligência Artificial (IA) é a área da ciência da computação responsável por desenvolver computadores e sistemas inteligentes (McKinnel *et al*., 2019). Para atingir tal objetivo, a IA se mantém como uma área dinâmica, em relação à pesquisa e ao desenvolvimento, desde seu nascimento na conferência de Dartmouth nos Estados Unidos em 1956 (CANTU-ORTIZ, 2014).

Segundo Talwar e Koury (2017), a inspiração para o desenvolvimento de computadores e sistemas inteligentes inclui aspectos como: reconhecimento de fala, tradução de idiomas, percepção visual, aprendizado, raciocínio, planejamento, tomada de decisão e intuição.

Naveen *et al*., (2019) apresentam em seu estudo os benefícios da aplicação de IA. São eles: sistemas mais poderosos que os sistemas e/ou algoritmos convencionais, capacidade de resolver novos problemas, capacidade de extrair conhecimento dos bancos de dados e baixa taxa de erro em comparação com os seres humanos.

Em muitos países, especialmente os desenvolvidos, incentiva-se a produção de estudos envolvendo a aplicação da IA em outras áreas. Isso faz com que outros países entendam a importância que a área de IA desempenha para o desenvolvimento da Ciência, Pesquisa e

Tecnologia. Com isso, pode-se dizer que o mundo entrou na era da IA, pois seu desenvolvimento está intimamente relacionado à modernização internacional e afeta diretamente os interesses dos países e de seus povos (MCKINNEL *et al*., 2019).

Dentro da área da IA, existem subáreas como o Processamento de Linguagem Natural (PLN), a Visão Computacional, o Aprendizado de Máquina e a Robótica (Naveen *et al*., 2019). Além disso, semelhante a outras áreas de pesquisa, a IA é caracterizada como um campo interdisciplinar por sua capacidade de ser aplicada em conjunto com outras áreas do conhecimento, como economia, administração e ciência da informação (VIJAYAKUMAR; SHESHADRI, 2019; MCKINNEL *et al*., 2019).

Segundo Mathews (2019), quando a IA é aplicada corretamente na área da segurança da informação, pode-se obter ganhos significativos, como um melhor desempenho em controles de segurança de um determinado sistema de informação. Mas para que a IA seja aplicada corretamente, é necessário definir primeiro o problema que se deseja resolver e, em seguida, selecionar a técnica mais apropriada para solucioná-lo.

2.5.1 Redes Neurais Artificiais

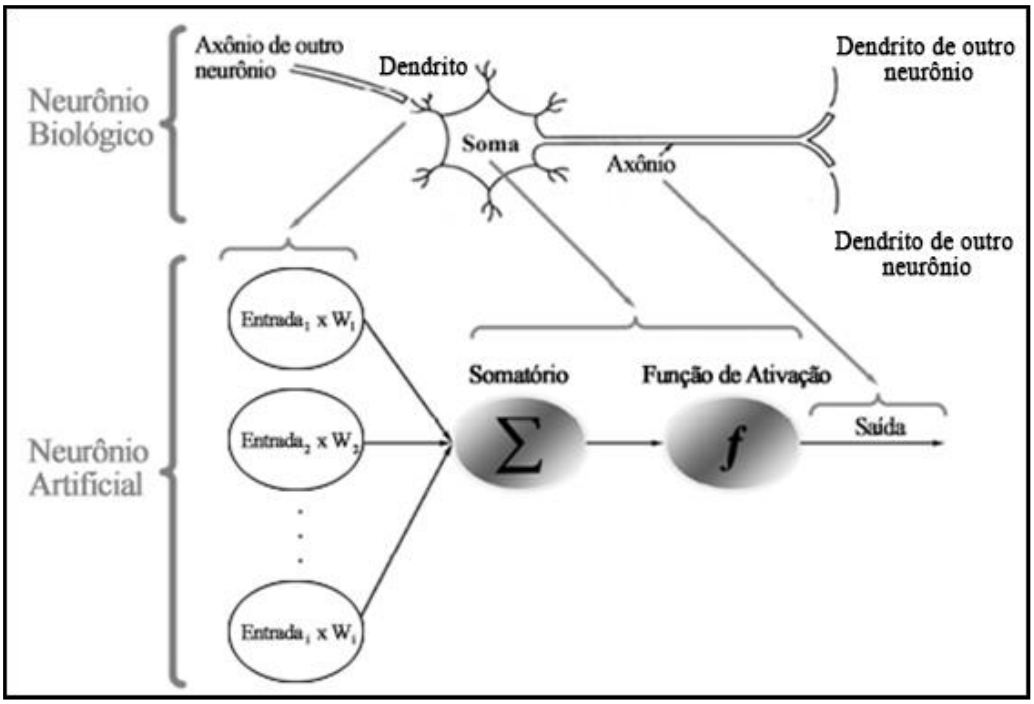
Uma técnica de IA que pode ser utilizada para solucionar problemas da área de segurança da informação são as Redes Neurais Artificiais (RNA). As RNAs podem ser utilizadas para diversas tarefas, como: Classificação, agrupamentos, associação, reconhecimento de padrões, regressão e predição (ABIODUN *et al*., 2018).

As RNAs são modelos matemáticos de inteligência artificial inspirados na estrutura do cérebro com o objetivo de simular o comportamento humano em processos como: aprendizado, adaptação, associação, generalização e abstração (HAYKIN, 2001).

Analisando-se a estrutura e o funcionamento de um neurônio biológico, pesquisadores tentaram simular esta estrutura e funcionamento em computadores. O modelo de neurônio artificial que foi consolidado e aceito pela comunidade científica foi proposto por Warren McCulloch e Walter Pitts em 1943.

Trata-se de um modelo matemático que implementa de forma simplificada os componentes e o funcionamento de um neurônio biológico (Heidari e Shamsi, 2019). Uma comparação entre o neurônio biológico e o neurônio artificial é apresentada na figura 9.

Figura 9 - Comparação entre o neurônio biológico e o neurônio artificial



Fonte: Traduzido de Heidari e Shamsi (2019)

A estrutura do neurônio biológico é constituída por três componentes: dendritos, soma ou corpo celular e axônios. Os dendritos recebem impulsos elétricos oriundos de outros neurônios e os conduzem até o soma, que processa os impulsos, gera outros e os envia pelo axônio para os próximos neurônios. A esta transmissão de impulsos elétricos é dada o nome de sinapse (HEIDARI; SHAMSI, 2019).

Nas RNAs, a aprendizagem se dá por meio de um conjunto de unidades simples de processamento chamadas de neurônios artificiais. Os neurônios artificiais são constituídos por: um vetor de dados de entrada X [x1, x2, x3....xn], os neurônios da camada de entrada, um vetor de pesos W [w1, w2, w3... wn] dos neurônios da camada de entrada, uma função de ativação e os neurônios da camada de saída.

No neurônio artificial, os dados dos vetores de entrada, os neurônios da camada de entrada e seus respectivos pesos são enviados para o soma, também chamado de junção aditiva representada pela letra sigma. Em seguida, a função de ativação do neurônio artificial é realizada de forma semelhante à sinapse no neurônio biológico, transmitindo ou bloqueando os impulsos nervosos. Desta forma, o aprendizado das RNAs acontece por meio dos ajustes dos pesos. O valor do peso será determinado em função do seu valor na iteração anterior, conforme demonstrado na Equação (1):

$$w_i^{t+1} = w_i^t + \Delta w_i^t \tag{1}$$

A atualização dos pesos depende do algoritmo, mas geralmente baseia-se na minimização do erro entre os valores previstos pela rede e as saídas desejadas, conforme pode-se observar na Equação (2):

$$\varepsilon_i = \sum w_i x_i - y_i \tag{2}$$

Desta forma, para que uma RNA execute tarefas como classificação e agrupamento, ela deve aprender. Para isto, fornece-se para a RNA um vetor de dados de entrada, também chamado de conjunto de exemplos de treinamento. Em seguida, inicia-se o vetor de pesos com valores aleatórios e calcula-se a saída da RNA. Ao longo do processo de treinamento, os valores do vetor de pesos são ajustados até que o erro tenha alcançado níveis baixos. Esse erro é utilizado como métrica para avaliar a qualidade da RNA (MITCHELL, 1997; HAYKIN, 1994).

Uma característica importante das RNAs é a capacidade de aprender de forma incompleta e sujeita a ruído. A tolerância a falhas faz parte da arquitetura devido à sua natureza distribuída de processamento. Se um neurônio falhar, sua saída incorreta será substituída pelas demais saídas corretas (HAYKIN, 2001).

Com a aplicação de uma RNA em segurança da informação, é possível obter resultados interessantes na classificação de sites maliciosos e *phishings* (Ferreira *et al*., 2018), no desenvolvimento de sistemas de detecção de intrusão (Amini; Jalili; Shahriari, 2006), na detecção de anomalias em sensores wireless (Siripanadorn; Hattagam; Teaumroong, 2010) e na classificação de tráfego normal e tráfego proveniente de ataques que exploram a vulnerabilidade de negação de serviço em um sistema de informação (CUI *et al*., 2019).

Os benefícios da aplicação da RNA na segurança da informação são apresentados por Wang, Lu e Qin (2020). Segundo os autores, além de a RNA recomendar as decisões mais apropiadas, ela também pode auxiliar para que a execução de testes e análises seja correta e automática. Dentre os principais benefícios da RNA, destacam-se a autoaprendizagem, a auto-organização, a alta tolerância a falhas, a robustez e o processamento paralelo.

As RNAs podem ser utilizadas para diferentes tipos de aprendizado, como o supervisionado e o não-supervisionado. No aprendizado supervisionado, consideram-se os dados de entrada e suas saídas correspondentes. Esse tipo de aprendizado é análogo ao aprendizado com um professor.

Os dados de saída servem como "professor" para o modelo, que utiliza a informação para adaptar sua estrutura e tornar-se capaz de generalizar e modelar novos dados. Para esse tipo de aprendizado, as tarefas mais comuns são a de classificação e a de regressão. Pode-se citar como exemplo de RNA com aprendizagem supervisionada o modelo Perceptron Multicamadas, ou *MultiLayer Perceptron* (MLP) (BAO; LIANJU; YUE, 2019).

Diferente da aprendizagem supervisionada, a aprendizagem não-supervisionada envolve inferir as propriedades dos dados sem a necessidade de um "professor". A tarefa mais comumente utilizada para esse tipo de aprendizado é o agrupamento, pois seu objetivo principal é formar grupos por similaridade. Quanto aos modelos de RNA para aprendizagem não-supervisionada, destacam-se Hopfield e os Mapas Auto-Organizáveis de Kohonen.

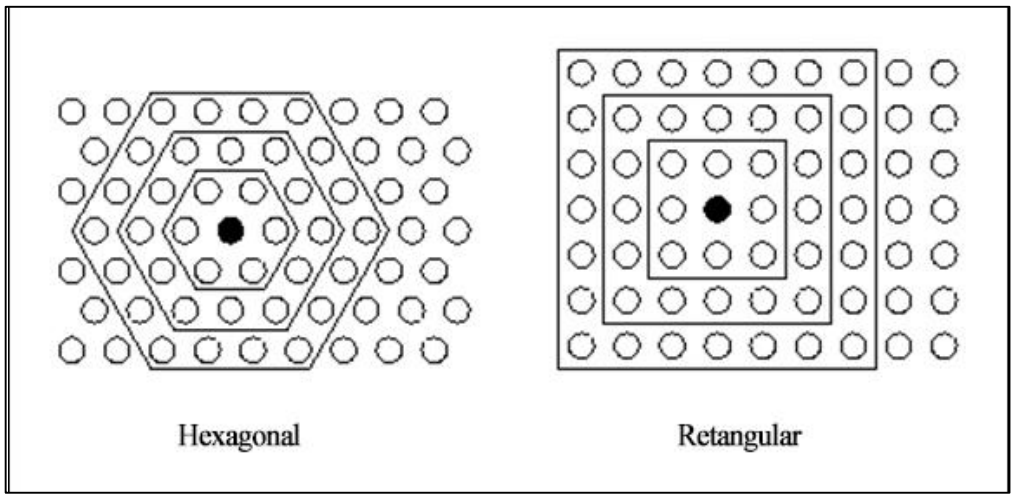
2.6 MAPAS AUTO-ORGANIZÁVEIS DE KOHONEN

Os Mapas Auto-Organizáveis de Kohonen, do inglês *Self-Organizing Maps* (SOM), são um tipo de RNA desenvolvida por Teuvo Kohonen, em 1984 (KOHONEN, 1984). Analisando-se a estrutura de outros modelos de RNA, a rede SOM é a que possui uma forte inspiração neurofisiológica. Sua estrutura é baseada no mapa topológico presente no córtex cerebral (KOHONEN, 1982).

A rede SOM é uma RNA baseada em aprendizagem não-supervisionada capaz de processar uma entrada de dados de um espaço multidimensional, transformando-a em um arranjo unidimensional ou bidimensional (KOHONEN, 1990).

A estrutura da rede SOM é composta por neurônios interconectados por uma relação chamada de vizinhança. É essa relação que determina a topologia do mapa. Em um mapa bidimensional, a vizinhança pode estar disposta de forma hexagonal ou retangular (KOHONEN, 1989). Na figura 10, os dois arranjos de vizinhança são exibidos.

Figura 10 – Arranjos hexagonal e retangular para uma rede SOM bidimensional



Fonte: Adaptado de Kohonen (1989).

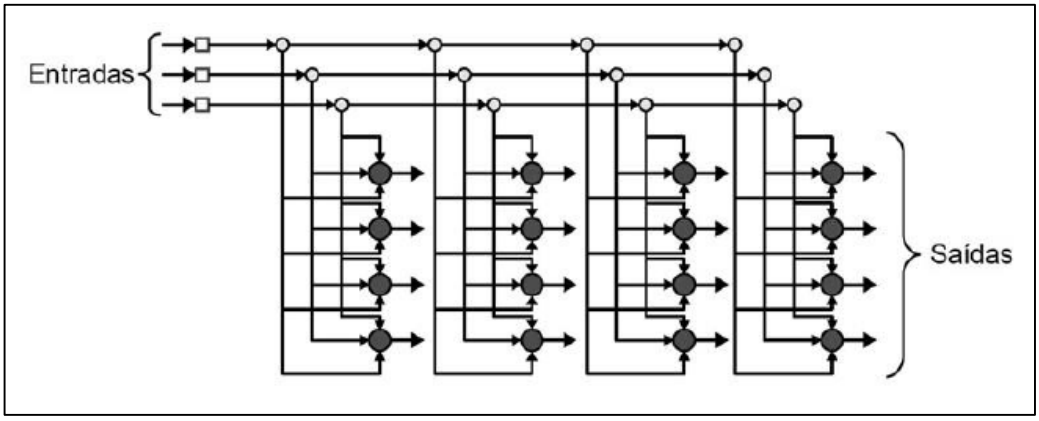
Na figura 10, pode-se observar, no centro dos arranjos hexagonal e retangular, o neurônio vencedor preenchido pela cor preta, bem como sua vizinhança imediata, que são seis neurônios no modelo hexagonal, e oito neurônios, no modelo retangular. O formato do arranjo dos neurônios influencia diretamente a adaptação da rede SOM aos dados de entrada, sendo o arranjo hexagonal o que oferece melhores resultados (KOHONEN, 1989).

O arranjo hexagonal oferece melhores resultados que o arranjo retangular, pois é o que melhor preserva e representa os dados do vetor de entrada. Isto acontece porque os vetores de peso tendem a aproximar a distribuição dos vetores de entrada no mapa de maneira ordenada (KOHONEN, 1989).

2.6.1 Algoritmo de Aprendizado da SOM

Para cada dado fornecido para a rede SOM, haverá uma competição entre todos os neurônios pelo direito de representá-lo. O neurônio que vencer a competição será o que possuir o vetor de pesos com os valores mais próximos do vetor de entrada. Esse tipo de aprendizado é chamado de aprendizado competitivo. Na figura 11, apresenta-se uma grade bidimensional de neurônios da rede SOM.

Figura 11 – Grade bidimensional de neurônios da rede SOM



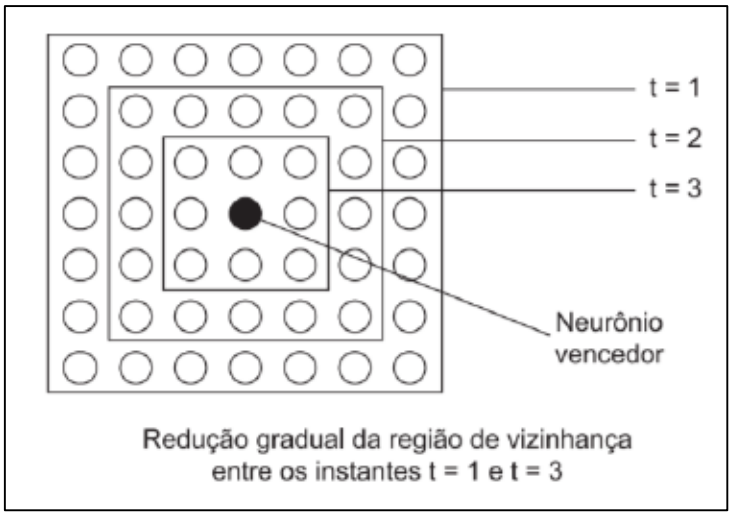
Fonte: Adaptado de Adaptado de Jarske, Seabra e Silva (2018).

No aprendizado competitivo, todos os neurônios da RNA recebem o mesmo vetor de dados de entrada. Em seguida, cada neurônio calcula seu nível de ativação multiplicando o seu vetor de pesos pelo vetor de entrada (LÓPEZ *et al*., 2019).

O neurônio que tiver o maior nível de ativação é chamado de "neurônio vencedor" ou de BMU (*Best-Matching Unit*). Desta forma, o vetor de entrada será representado por um único neurônio ou por grupos, sendo cada um deles representado por um neurônio (KANGAS; KOHONEN; LAAKSONEN, 1990).

Após a definição do BMU, uma área é definida ao seu redor, onde os neurônios com os valores de saída mais próximos são posicionados (KANGAS; KOHONEN; LAAKSONEN, 1990). Essa relação entre os neurônios, chamada de "vizinhança", é apresentada na Figura 12.

Figura 12 – Relação de vizinhança entre os neurônios em um arranjo hexagonal



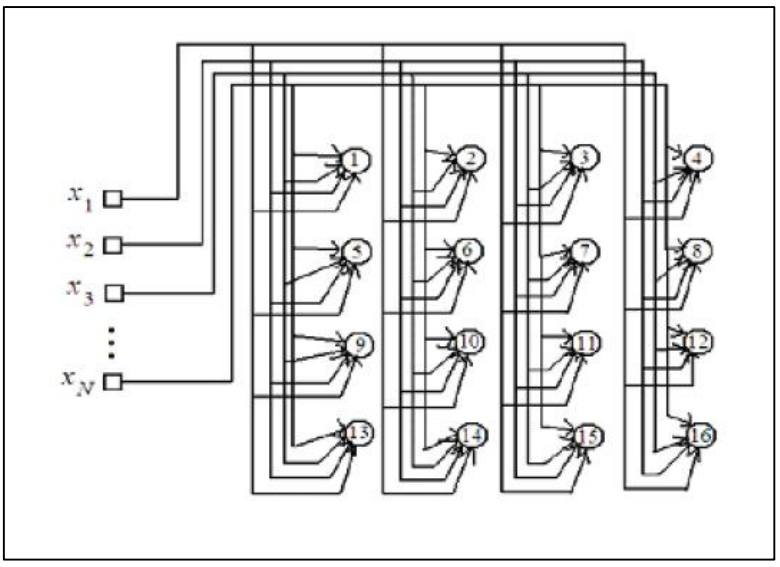
Fonte: Adaptado de Kangas, Kohonen e Laaksonen (1990) e Ferreira (2019).

Na Figura 12, está representada a relação de vizinhança para a formulação do BMU e seu respectivo grupo. Inicialmente, o grupo dos neurônios é representado pelo “raio da vizinhança” inicial, o Nc(t1). Esse primeiro raio de vizinhança é amplo e cobre quase metade do mapa. No final do processo, apenas os neurônios com os valores mais próximos ao BMU fazem parte de sua vizinhança, formando um agrupamento, conforme o Nc(t3). Isto permite que sejam gerados mapas globalmente ordenados, afinal, se o mapa fosse gerado apenas com o primeiro grupo Nc(t1), o resultado seria um mapa muito amplo e desordenado (KANGAS; KOHONEN; LAAKSONEN, 1990).

Durante a fase de treinamento da rede SOM, os vetores de saída são constantemente adaptados às informações fornecidas pelo vetor de entrada X. Para cada atualização, o vetor de entrada é atribuído ao melhor neurônio correspondente (o mais similar), bem como ao subconjunto de neurônios vizinhos. Isto permite que a rede SOM forneça uma redução de dimensionalidade e um agrupamento dos dados multidimensionais (SASSI, 2006; SACHA *et al.*, 2017).

Na figura 13, apresenta-se um exemplo da fase de treinamento da rede SOM, simulando 16 neurônios recebendo simultaneamente o vetor de entrada X.

Figura 13 – Aprendizado competitivo em uma rede SOM com 16 neurônios



Fonte: Adaptado de Kohonen (1997).

Assim, quando cada um dos vetores de entrada **X** é processado pela rede SOM, cada um dos neurônios de saída recebe um valor e calcula seu nível de ativação, conforme demonstrado na equação (3),

$$u_i = \sum_{k=1}^{N} w_{ik} x_k \ , \ i=1,...,M, \tag{3}$$

Sendo que X é o vetor de entrada, i é o índice que indica qual é o neurônio que está recebendo o valor de entrada e w_i é o vetor de pesos entre o valor de entrada e do neurônio. O BMU será o neurônio que possuir o maior valor de u_i, ou seja, o que mais se aproximar do vetor de entrada. Esse será o neurônio que irá representar o padrão dos dados do vetor de entrada. Os demais neurônios **M** competem para determinar qual deles receberão um valor mais próximo do BMU para também permanecerem ativos. Essa competição faz com que esse tipo de rede também seja chamado de "O vencedor fica com tudo" ("*Winner-takes-all*").

Segundo Haykin (1994), pode-se sintetizar o algoritmo da rede SOM em cinco etapas, que são descritas na Tabela 8.

Tabela 8 – Cinco etapas da SOM sintetizadas por Haykin

Etapa	**Descrição**
Inicialização	Escolha dos valores aleatórios para os vetores de pesos.
Escolha do Padrão de Entrada	Escolha de um padrão x de neurônios e a determinação de sua vizinhança.
Determinação do Neurônio Vencedor	Escolha do neurônio vencedor baseando-se na similaridade entre o nível de ativação do neurônio e o valor de entrada.
Atualização dos Pesos	Modificação dos valores dos vetores dos pesos dos neurônios da rede.
Continuação	Repetição dos passos 2,3 e 4 até que não sejam observadas mudanças significativas no mapa.

Fonte: Adaptado de Haykin (1994)

2.6.2 Medidas de Qualidade da SOM

Para medir a acurácia do mapa e da dimensão escolhida, ou seja, medir a qualidade do mapa e analisar se a topologia escolhida é a que "melhor representa os dados do vetor de entrada X", algumas medidas de qualidade podem ser utilizadas, como o Erro de Quantização (EQ) e o Erro Topográfico (ET) (Kohonen, 2001). Na tabela 9, apresenta-se a descrição de cada uma dessas medidas.

Tabela 9 – Medidas de acurácia para a SOM

Medida	**Descrição**	**Equação**
Erro de Quantização	Mostra a qualidade dos dados do vetor de entrada. Quanto melhor a qualidade do vetor de entrada, melhor será a disposição dos neurônios no mapa. O erro de quantização será próximo de zero quando todos os nós estiverem bem distribuídos no mapa.	$EQ = \frac{1}{N}\sum_{k=1}^{N}\left\|\boldsymbol{x} - \boldsymbol{w}_{i^*}\right\|$
Erro Topográfico	Mede a preservação da topologia dos dados de entrada. Como os dados estão indo de um espaço multidimensional para um espaço bidimensional ou unidimensional, eles acabam perdendo informação. Uma forma de avaliar a representação do vetor de entrada inicial é utilizando o erro topográfico. Quando o erro topográfico for **próximo** de zero, significa que todos os nós representam bem o vetor de entradas inicial.	$ET = \frac{1}{N}\sum_{k=1}^{N}\boldsymbol{u}(\boldsymbol{x}_k)$

Fonte: Adaptado de Kohonen (2001).

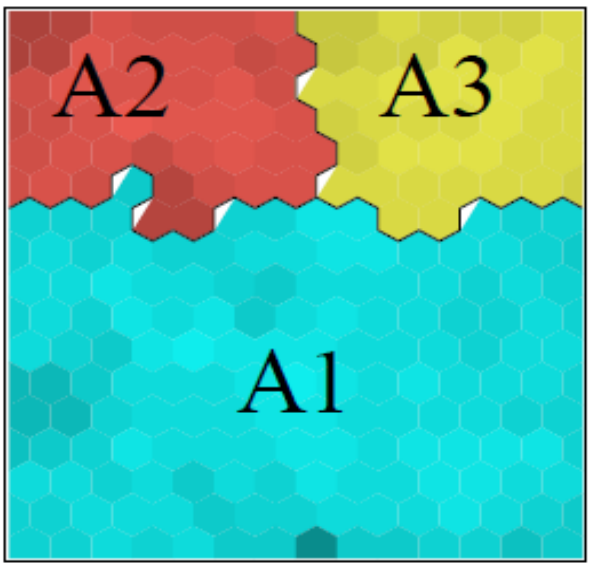
2.6.3 Dimensão e Visualização do Mapa Gerado Pela Som

Quanto à dimensão do mapa gerado pela rede SOM, ou seja, a quantidade de neurônios utilizados para representar o vetor de entrada, pode-se variar de dezenas a centenas de neurônios. Não é possível predizer corretamente o tamanho exato do mapa, mas ele deve ser capaz de representar a relação dos dados do vetor de entrada. Uma forma de determinar a dimensão do mapa é pelo método de tentativa e erro (KOHONEN; SOMERVUO, 1998; KOHONEN, 2013).

O mapa gerado pela rede SOM contém os neurônios agrupados por similaridade. Cada um deles possui um BMU, ou seja, um vetor de saída que representa os demais neurônios contidos no agrupamento (KOHONEN; SOMERVUO, 1998).

Na Figura 14, é apresentado um exemplo de mapa gerado pela rede SOM, com 3 agrupamentos, feito pelo autor Ferreira (2019). O mapa foi gerado em uma base de dados sobre absenteísmo no ambiente de trabalho. Tal base pode ser acessada na UCI *Machine Learning Repository,* por meio do link: http://archive.ics.uci.edu/ml/datasets/Absenteeism+at+work.

Figura 14 – Mapa com três agrupamentos



Fonte: Ferreira (2019).

Em relação à aplicação da rede SOM na área de segurança da informação, há estudos que mostram resultados interessantes, como é o caso da investigação forense em falhas de softwares (BELLA; ELOFF, 2016), da detecção de anomalias em sites on-line (LEE; KIM; KIM, 2011), da identificação de padrões para melhorar a reputação de sistemas de informação (BANKOVIC *et al*., 2011), da identificação do comportamento dos usuários nos incidentes e nas fraudes de segurança da informação (LÓPEZ *et al*., 2019), da detecção de intrusão de ataques DDoS (LI, 2007), da identificação de ameaças em classificadores de sinais (CLANCY; KHAWAR, 2009) e da padronização de categorias de vulnerabilidades (VENTER; ELLOF; LI, 2008).

2.7 PROCESSAMENTO DE LINGUAGEM NATURAL

Com o crescimento significativo de conteúdo gerado por usuários na internet, a extração automática de informações relevantes passou a receber interesse de pesquisadores de diversas áreas, sendo que muitos deles estão conseguindo esse feito por meio do Processamento de Linguagem Natural (PLN) (SUN; LUO; CHEN, 2017).

O Processamento de Linguagem Natural (PLN), do inglês "*Natural Language Processing*", é a subárea da IA responsável por fazer com que os computadores interpretem e desenvolvam conteúdo em linguagem humana. Por se tratar de uma área interdisciplinar, a IA inclui outras áreas como: Ciência da Computação, Linguística, Psicologia e Estatística (NOUBOURS; PRITZKAU; SCHADE, 2014).

A aplicação de PLN em textos ou em outros conteúdos de linguagem humana pode ser executada por meio de diversas tarefas. Entre elas, destacam-se: stemização, produção de corpus, tokenização, lematização, marcação gramatical, análise sintática (*Parsing*), concordância, frequência, colocação estatística e remoção de *stopwords* (VIJAYAKUMARA; FUAD, 2019; ZEROUAL; LAKHOUAJA, 2018). As principais tarefas de PLN são descritas na Tabela 10.

Tabela 10 – Principais tarefas de PLN

Tarefa	**Descrição**
Stemização	Utilizada para consolidar diferentes variações de palavras que compartilham o mesmo radical em uma forma raiz comum. Por exemplo, as palavras “Gosto” e “Gostei” serão simplificadas para a forma raiz de "Gost".
Corpus	Trata-se da formação de um conjunto de todas as palavras presentes em um texto em um único item. Também chamado de “Base de Texto”, é utilizado na maioria das tarefas de PLN.
Tokenização	A tokenização, também chamada de “Segmentação de Palavras”, é responsável por quebrar uma determinada sequência de caracteres de um texto, ou seja, ela determina onde as palavras de um texto iniciam e terminam e as transformam em tokens. Os tokens são listas geradas a partir de um corpus tokenizado.
Lematização	Redução de palavras superficiais à sua forma canônica chamada lema. O lema relaciona diferentes formas de palavras com o mesmo significado. Por exemplo, a palavra “Melhor” tem a palavra “Bom” como seu lema. Sua utilização é eficiente, em particular, para recuperação de informações.
Marcação Gramatical (POS)	Trata-se de uma tarefa básica na linguística de corpus. O objetivo é atribuir características morfossintáticas a cada palavra em uma frase de acordo com o seu contexto. Essa tarefa também pode ser aplicada em sentenças e parágrafos.
Análise Sintática	Sucessora natural da marcação gramatical, a análise sintática fornece uma árvore de dependência como saída de cada palavra dentro de um corpus. Seu objetivo é prever, para cada sentença ou cláusula, uma representação abstrata das entidades gramaticais e suas relações.
Remoção de *Stopwords*	A remoção de *stopwords* tem o objetivo de manter um corpus mais conciso e limpo para futuras análises. Um exemplo de aplicação é a remoção de palavras como: “de”, “se”, são” e “é”.
Concordância	Tem como objetivo pesquisar em um corpus todas as ocorrências de cada palavra selecionada e exibir seu contexto imediato. Além disso, as concordâncias podem ser produzidas em vários formatos, mas a forma mais comum é a concordância para produzir palavras-chave.
Frequência	Utilizado para produzir listas de palavras e sua frequência em um determinado corpus. Além disso, é possível produzir listas de frequência de palavras usando um corpus marcado com marcação gramatical.
Colocação Estatística	É um procedimento utilizado para calcular informações estatísticas sobre a associação das palavras, a força da colocação e as frequências comparativas de formas de palavras em um corpus.

Fonte: Adaptado de Vijayakumara e Fuad (2019); Zeroual e Lakhouaja (2018).

Sun, Luo e Chen (2017) apresentam as principais bibliotecas e ferramentas utilizadas para implementar e desenvolver algoritmos, além das principais técnicas de IA usadas para PLN. Na Tabela 11, as principais bibliotecas e ferramentas são descritas.

Tabela 11 – Principais bibliotecas e ferramentas para PLN

Biblioteca	**URL**	**Descrição**
NLTK	http://www.nltk.org	Biblioteca de código aberto utilizada para executar tarefas de tokenização, mineração de opiniões, análise de sentimentos e raciocínio semântico.
OpenNLP	https://opennlp.apache.org/	Biblioteca utilizada para processamento de textos. Suporta tarefas como segmentação de sentença, reconhecimento de entidades e também análise de sentimentos.
CoreNLP	http://stanfordnlp.github.io/CoreNLP/	Biblioteca com capacidade para análises avançadas de sentimentos.
Gensim	http://radimrehurek.com/gensim/	Biblioteca de código aberto utilizada para modelar tópicos que abordem a Análise Semântica Latente (LSA) e a Alocação de Dirichlet Latente (LDA)
Fudan NLP	https://code.google.com/archive/p/fudannlp/	Kit de ferramentas de código aberto para PNL chinesa. Suporta tarefas como: segmentação de palavras, marcação de POS, nomeação de entidades e análise de dependências.
LTP	http://www.ltp-cloud.com/intro/en/	Sistema de código aberto para linguagem chinesa, incluindo análise lexical, análise sintática e análise semântica.
NiuParser	http://www.niuparser.com	Kit de ferramentas de análises sintática e semântica para a língua chinesa. Suporta tokenização, marcação de POS, análise de dependência e rotulação de funções semânticas.

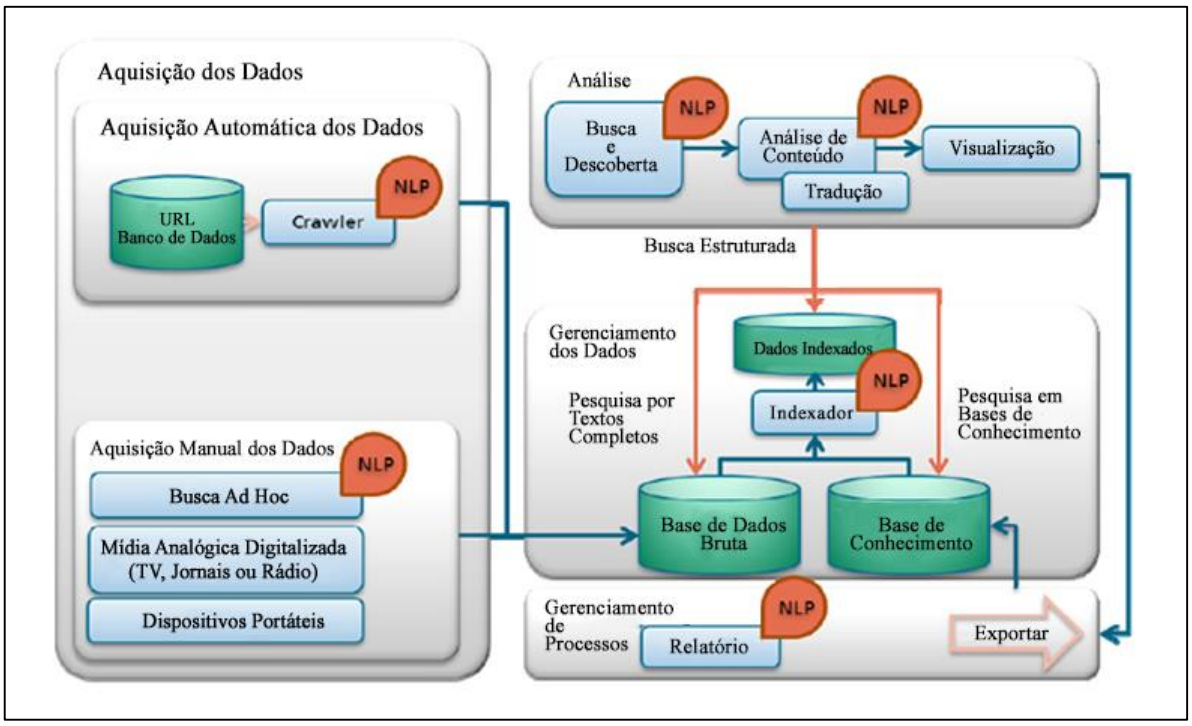
Fonte: Adaptado de Sun, Luo e Chen (2017).

Na área de segurança da informação, estudos mostram uma tendência de aplicação de PLN em Pentest, principalmente na fase inicial, chamada “Reconhecimento” ou “OSINT”. A justificativa é que a aplicação de PLN aumenta a eficácia na descoberta de vulnerabilidades já publicadas e documentadas, tais como: versões de softwares desatualizadas e arquivos de configurações de dispositivos on-line (YOU *et al.*, 2017).

Quanto ao uso de PLN em conjunto com o OSINT, segundo Noubours, Pritzkau e Schade (2014), aumenta-se o desempenho em algumas tarefas, como, por exemplo, na aquisição automática e manual dos dados, na análise e descoberta de dados, no momento da busca por informação e no gerenciamento dos resultados encontrados.

Na figura 15, apresentam-se algumas tarefas executadas por OSINT, em que o PLN pode ser aplicado.

Figura 15 – Aplicações de PLN em OSINT



Fonte: Traduzido de Noubours, Pritzkau e Schade (2014).

Conforme demonstrado na Figura 15, são apresentadas a seguir as tarefas executadas por OSINT, em que o PLN foi aplicado.

- **Aquisição dos dados**: Para realizar a coleta dos dados, deve-se primeiramente estabelecer um plano de execução, que precisa conter as ferramentas, os *frameworks* ou as abordagens que serão utilizadas, bem como as possíveis fontes abertas onde a informação possa estar armazenada. Abordagens gerais para aquisição de dados se diferenciam entre automático e manual. A execução automática é geralmente mais rápida e eficaz que a manual. Sua execução é realizada por mecanismos de busca ou por ferramentas que contenham algum componente de um mecanismo de busca, como um "rastreador". Utilizar técnicas de PLN na aquisição dos dados possibilita a análise automática do conteúdo dos sites antes da realização de outra tarefa, como uma análise sintática dos resultados.

- Gerenciamento dos dados: Na execução de OSINT, informações de diversos tipos e vindas de fontes abertas precisam ser mescladas, processadas e analisadas, o que é uma tarefa árdua, pois os dados oriundos de fontes abertas, em sua grande maioria, são não-estruturados. Os dados estruturados são informações que estão organizadas em uma determinada estrutura ou modelo, permitindo que seu conteúdo seja pesquisável mais facilmente. Como grande parte

do conteúdo disponível na internet, e em outras fontes abertas, são não-estruturadas, é importante a aplicação de PLN para lidar com o volume de dados.

- **Análise:** Para que a execução de OSINT seja feita com sucesso, a informação coletada deve ser processada, analisada e mesclada. O resultado ideal de uma execução de OSINT é uma informação que possa ser estruturada e que acrescente um novo conhecimento. Para isso, em adição ao uso de NLP, recomenda-se a aplicação de técnicas de análise de dados, como pesquisa e filtro, pré-processamento, transformação dos dados e visualização.

- **Gerenciamento de Processos:** Após análise, o responsável pela execução de OSINT necessita revisar e documentar os resultados obtidos. Isso inclui resumir e relatar as descobertas por meio de relatórios ou apresentações. Para o gerenciamento de processos, recomenda-se a utilização de ferramentas capazes de produzir uma visualização dos resultados encontrados, uma vez que a visualização permite um melhor entendimento das observações apontadas, e o desenvolvimento das próximas perguntas a serem feitas, ou seja, a continuidade da execução de OSINT. O gerenciamento de processos é uma das etapas mais importantes, pois é quando o responsável pela execução de OSINT compartilha e discute seus achados.

Considerando o aumento de desempenho que o PLN proporciona na execução de OSINT, os autores Noubours, Pritzkau e Schade (2014) recomendam a utilização de outras técnicas de IA em conjunto com PLN. Desta forma, torna-se possível extrair conhecimento das informações obtidas por OSINT.

3 MATERIAIS E MÉTODOS

Neste capítulo, abordam-se os materiais e os métodos utilizados para realização deste trabalho, assim como as fases dos experimentos computacionais.

3.1 CARACTERIZAÇÃO METODOLÓGICA

A metodologia de pesquisa utilizada neste trabalho é definida, com base em sua natureza, como pesquisa aplicada, pois tem o objetivo de gerar um novo conhecimento para solucionar um problema do mundo real, possuindo, assim, uma aplicação prática (GIL, 2008).

Este trabalho também é caracterizado como uma pesquisa quantitativa que, segundo Gerhardt e Silveira (2009), utiliza diferentes técnicas estatísticas para quantificar os resultados obtidos. Esse tipo de pesquisa recorre à linguagem matemática para descrever as causas de um fenômeno, as relações entre variáveis, dentre outros fatores.

O trabalho aqui apresentado é classificado também como descritivo. A pesquisa descritiva tem como objetivo descrever as características de uma determinada população/fenômeno ou estabelecer relações entre variáveis. Suas características mais significativas são o uso de técnicas para a coleta de dados e análise dos resultados (GIL, 2008).

Quanto aos procedimentos técnicos, esta pesquisa é do tipo experimental. Segundo Gil (2008), a pesquisa experimental consiste em determinar um objeto de estudo e selecionar as variáveis que podem influenciá-lo. Para isso, é preciso definir as formas de controle e observação dos efeitos que as variáveis produzem no objeto de estudo determinado.

Na Seção 3.4 deste capítulo, realizou-se uma revisão sistemática da literatura com diferentes materiais publicados, tais como livros, teses, sites e artigos científicos, para familiarizar o autor com o tema de pesquisa e, assim, ajudá-lo a compreender o atual cenário e a esclarecer ou modificar conceitos (GIL, 2008).

3.2 BASE DE DADOS E PLATAFORMA DE ENSAIOS

A base de dados selecionada foi a Google Hacking Database (GHDB). Tal escolha se deu pelo fato de essa base possuir a maior quantidade de Dorks documentadas e testadas dentre todas as que estão disponíveis na internet (ZHANG, NOTANI; GU, 2015; MEUCCI; MULLER, 2014). A GHDB encontra-se disponível no seguinte endereço eletrônico:

https://www.exploit-db.com/google-hacking-database. Além disso, ela possui um total de 4.211 Dorks e 4 atributos: Data: contém a data que a Dork foi publicada na Base, Dork: contém a Dork e seu link de acesso, Categoria: informa a qual categoria a Dork pertence e Autor: informa quem enviou a Dork para a base.

Na Tabela 12, apresenta-se uma amostra da base GHDB.

Tabela 12 – Amostra do GHDB

Data	Dork	Categoria	Autor
12/08/2019	intitle:Administration - Installation - MantisBT	Footholds	Mr.XSecr3t
14/06/2018	"username.xlsx" ext:xlsx	Files Containing Usernames	ManhNho
22/08/2019	intitle:"index of" /content/admin/	Sensitive Directories	Reza Abasi
02/01/2019	"dispatch=debugger."	Error Messages	deadroot

Fonte: o autor (2020).

Por se tratar de uma base on-line, foi necessário copiar as Dorks do site e exportá-las para um arquivo .CSV. As características do GHDB são apresentadas neste trabalho no Capítulo 2, Seção 2.4.1. Na Tabela 13, são descritas as categorias da base de Dorks utilizada neste trabalho.

Tabela 13 – Categorias de Dorks do GHDB

Categorias do GHDB	Quantidade de Dorks
Footholds	89
Files Containing Usernames	20
Sensitive Directories	268
Web Server Detection	125
Vulnerable Files	58
Vulnerable Servers	94
Error Messages	105
Files Containing Juicy Info	433
Files Containing Passwords	212
Sensitive Online Shopping Info	9
Network of Vulnerability Data	72
Pages Containing Login Portals	398
Various Online Devices	349
Advisories and Vulnerabilities	1979
Total	4211

Fonte: o autor (2020).

A base possui um total de 14 categorias, totalizando 4.211 Dorks. Dentre todas as categorias, a que possui maior quantidade de Dorks é a “Advisories and Vulnerabilities”, com 1.979 Dorks.

Descrevem-se, na Tabela 14, os softwares utilizados, juntamente com sua descrição, URL e sua utilização neste trabalho.

Tabela 14 – Softwares utilizados

Software	URL	Descrição	Utilização
Spyder	https://www.spyder-ide.org/	IDE para desenvolvimento em Python	Desenvolvimento e aplicação de PLN na GHDB
Mendeley	https://www.mendeley.com/	Software para gerenciamento de artigos	Criação do arquivo de referências bibliográficas utilizado no VosViewer
Vosviewer	https://www.vosviewer.com/	Software para construção e visualização de redes bibliográficas	Criação dos mapas de palavras-chave da revisão sistemática da literatura realizada neste trabalho
NLTK	https://www.nltk.org	Biblioteca em Python para PLN	Aplicação de PLN na GHDB
Pandas	https://pandas.pydata.org	Biblioteca para análise de dados em Python	Manipulação da estrutura da GHDB
Numpy	https://numpy.org/	Biblioteca para computação científica em Python	Geração de variáveis para apoiar a aplicação de PLN
Excel	https://office.live.com/start/Excel.aspx?ui=pt-BR	Planilha eletrônica	Utilizada para preencher os valores nulos no GHDB e para gerar os gráficos da revisão sistemática da literatura
Viscovery SOMine	https://www.viscovery.net/somine/	Software para aplicação da rede SOM	Aplicação de SOM no GHDB
Microsoft Visio 2016	www.buysoft.com.br/microsoft/visio	Criação de fluxogramas e diagramas	Desenvolvimento da abordagem proposta neste trabalho
Oracle VirtualBox	https://www.virtualbox.org/	Criação de máquinas virtuais	Criação da máquina virtual para testar as ferramentas de execução de Dorks
ParrotSec	https://parrotlinux.org/	Sistema operacional para Pentest	Sistema operacional utilizado para instalar e testar as ferramentas para execução automática de Dorks
Carrot2	http://search.carrot2.org/stable/search	Mecanismo de metabusca on-line	Busca de ferramentas que executem Dorks de forma automática
Google	http://google.com	Mecanismo de busca on-line	Busca de ferramentas que executem Dorks de forma automática
DorkMe	https://github.com/blueudp/DorkMe	Ferramenta para execução automática de Dorks	Execução das Dorks automaticamente
R Studio	https://rstudio.com/	IDE de desenvolvimento da linguagem R	Utilizada para desenvolver o Código em R
SOMbrero	https://cran.r-project.org/web/packages/SOMbrero/index.html	Biblioteca em R para aplicação da rede SOM	Utilizada para calcular o erro topográfico da rede SOM

Fonte: o autor (2020).

Os experimentos computacionais foram realizados em um Notebook Lenovo Ideapad320 com as seguintes configurações:

- Processador Intel Core I5-7200U.
- 8 Gigabytes de memória RAM.
- Disco rígido de 1 Terabyte de armazenamento.
- Placa de vídeo Nvidia Geforce 940Mx com 2 Gigabytes de memória de vídeo.

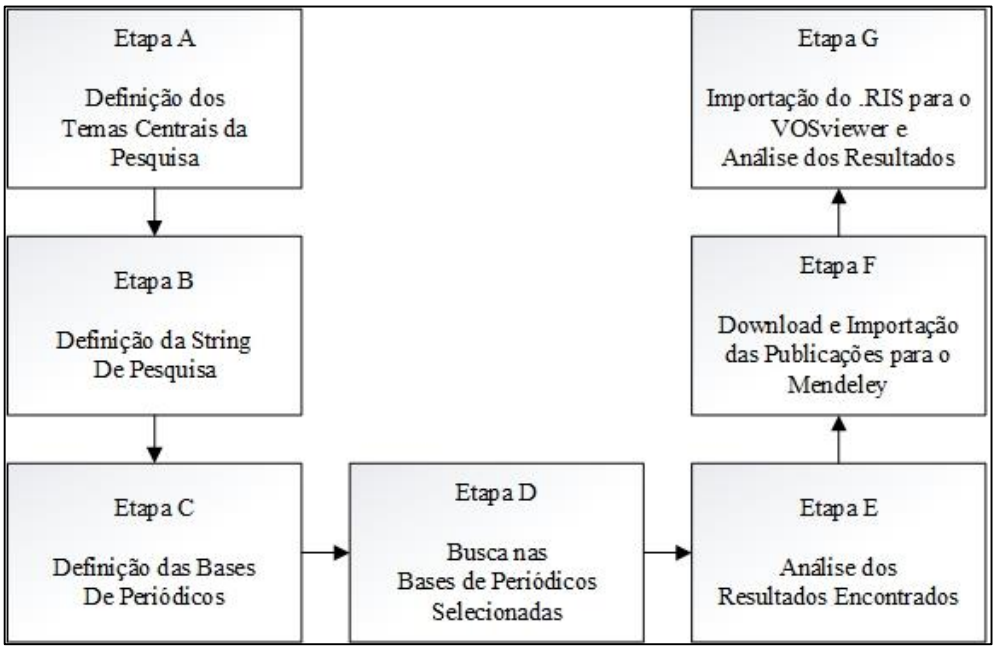
3.3 REVISÃO SISTEMÁTICA DA LITERATURA

A Revisão Sistemática da Literatura (RSL) é um método de pesquisa cientifica que tem como objetivos, elaborar uma questão de pesquisa, entender o cenário do tema escolhido em bases de publicações, delimitar critérios de inclusão e exclusão de publicações; e avaliar a qualidade metodológica das publicações selecionadas (KITCHENHAM; CHARTERS, 2007).

Neste trabalho, optou-se por realizar uma Revisão Sistemática da Literatura para evidenciar as tendências de aplicação de OSINT e confirmar a lacuna de pesquisa, em se tratando de publicações.

Realizou-se uma revisão sistemática da literatura dividida em sete etapas, como pode-se observar na Figura 16.

Figura 16 – As sete etapas da revisão sistemática da literatura



Fonte: o autor (2020).

Descrevem-se, a seguir, as sete etapas da revisão sistemática da literatura.

Etapa A (Definição dos Temas Centrais da Pesquisa): A primeira etapa foi definir os temas centrais da pesquisa. Baseando-se no tema deste trabalho, os temas escolhidos foram: "OSINT", "Open Source Intelligence" e "Inteligência de Fontes Abertas". Optou-se por pesquisar o termo OSINT em português para identificar o contexto nacional da pesquisa acadêmica sobre o tema. O termo em inglês por extenso foi escolhido para tentar encontrar publicações que abordem o OSINT, mas não usam seu acrônimo.

Etapa B (Definição da *String* de Pesquisa): Nesta etapa, definiu-se a *string* de consulta, construída com os termos centrais escolhidos anteriormente. A *string* de pesquisa definida foi: "OSINT" OR "Inteligência de Fontes Abertas" OR "Open Source Intelligence". Utilizando essa *string*, buscou-se encontrar publicações sobre OSINT em inglês e português, bem como publicações contendo seu acrônimo.

Etapa C (Definição das Bases de Periódicos): As bases de periódicos definidas foram: "ACM Digital Library", "Emerald Insight", "IEEEXplore – Digital Library", "ScienceDirect" e "Portal Capes", que inclui outras bases de periódicos como "Scopus", "ProQuest", "Web of Science" e "Google Scholar".

Para a seleção das publicações, os seguintes critérios foram definidos: ser artigo ou capítulo de livro, além de abordar os temas "OSINT", "Inteligência de Fontes Abertas" ou "Open Source Intelligence".

Etapa D (Busca nas Bases de Periódicos Selecionadas): A busca foi efetuada considerando publicações realizadas no período entre janeiro de 1990 e fevereiro de 2020. Encontrou-se um total de 5.04 publicações, como pode-se observar na Tabela 15.

Tabela 15 – Resultado da busca efetuada nas bases de periódicos selecionadas

Base	**Selecionados**	**Desconsiderados**	**Total**
ACM Digital Library	9	84	93
Emerald Insight	25	14	39
IEEEXplore – Digital Library	94	29	123
Portal Capes	46	83	129
ScienceDirect	74	46	120
Total	**248**	**256**	**504**

Fonte: o autor (2020).

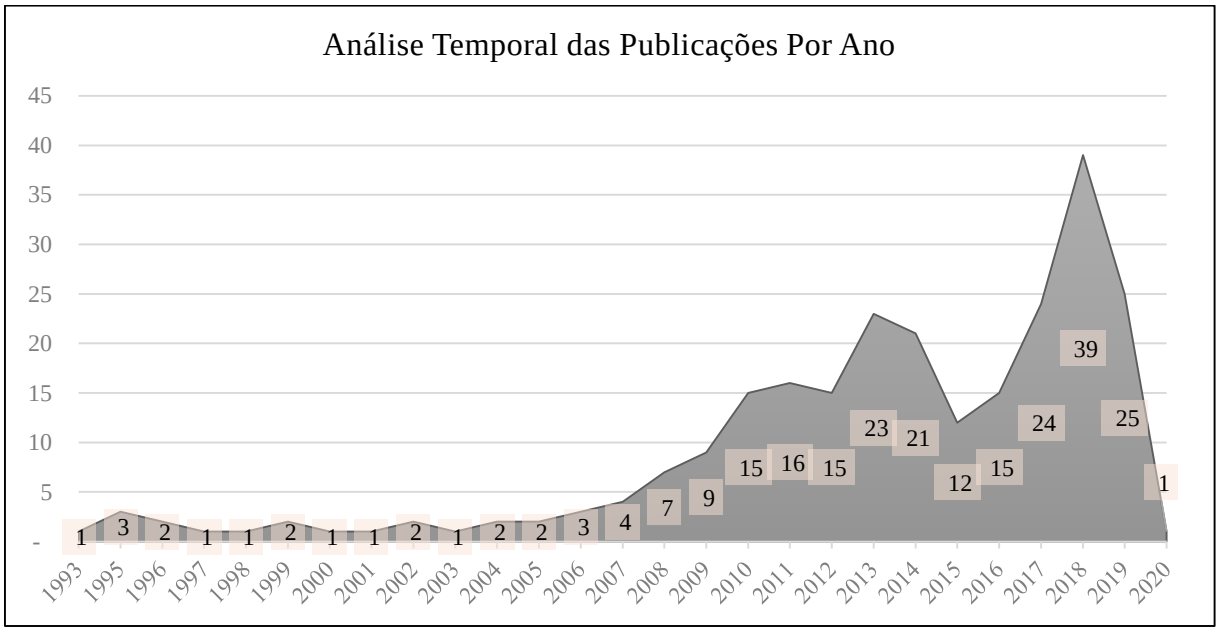
Analisando os resultados da Tabela 15, pode-se observar que as bases de periódicos com maior concentração de publicações sobre OSINT são: Portal Capes, ScienceDirect e IEEExplore – Digital Library. Em adição, outra base que trouxe um número interessante de publicações foi a ACM Digital Library; porém, como esta base, assim o Portal Capes, possuem publicações duplicadas em outras bases, o número de publicações desconsideradas foi superior ao número de publicações selecionadas.

Etapa E (Análise dos Resultados Encontrados): Das 504 publicações encontradas na Etapa D, 256 foram desconsideradas por: (1) não abordarem o tema OSINT, (2) serem publicações duplicadas em diferentes bases de periódicos, (3) serem índices ou guias de publicações e (4) terem as palavras-chave sobre OSINT apenas nas referências. Assim, 248 publicações foram selecionadas.

Em seguida, realizou-se a análise temporal para identificar em qual período está a maior concentração de publicações sobre OSINT, sendo que a primeira publicação foi realizada em 1993, totalizando 27 anos de pesquisa até a última publicação feita em fevereiro de 2020, conforme identificado na revisão sistemática deste trabalho. Além disso, foi possível observar que, a partir de 1995, todos os anos seguintes possuem ao menos uma publicação sobre OSINT, o que demonstra que o tema não deixou de ser estudado.

Com a análise dos resultados encontrados, verificou-se que a maior concentração de publicações ocorreu entre 2010 e 2019, e o ano com maior quantidade de trabalhos publicados sobre o tema OSINT foi em 2018, com um total de 39 publicações (Figura 17). Vale ressaltar que a pesquisa foi realizada até fevereiro de 2020, o que pode refletir o pequeno número apontado neste ano.

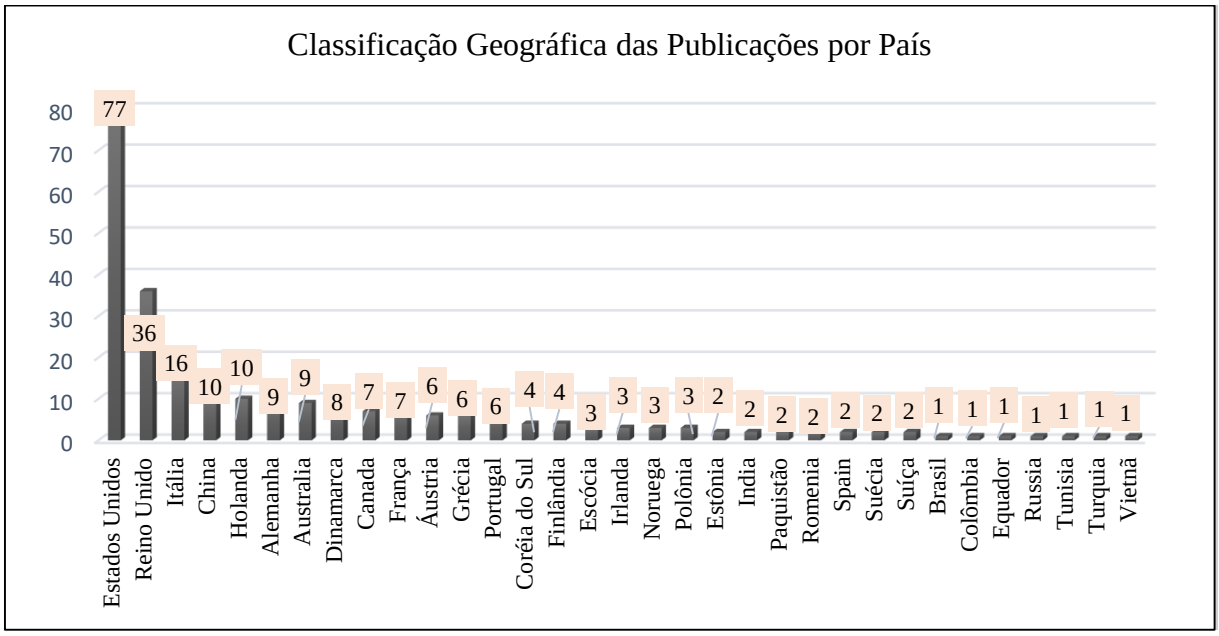
Figura 17 – Análise temporal das publicações sobre OSINT



Fonte: o autor (2020).

Além da análise temporal, realizou-se também a classificação geográfica, o que possibilitou identificar quais são os países e os continentes que mais publicam sobre OSINT. Para tal constatação, verificou-se a nacionalidade dos autores da publicação. No caso de publicações com autores de países diferentes, considerou-se o país do primeiro autor. Na Figura 18, mostra-se a classificação geográfica das publicações sobre OSINT por país.

Figura 18 – Classificação geográfica das publicações por país



Fonte: o autor (2020).

De todas as publicações encontradas na revisão sistemática da literatura, apenas 33 países publicaram algum estudo sobre o OSINT, sendo que o maior número de publicações está concentrado nos Estados Unidos, com um total de 77 publicações. Os outros países que possuem uma quantidade maior de publicações são: Reino Unido, com 36, Itália, com 16, e China e Holanda, com 10 cada.

Esse grande volume de publicações nos Estados Unidos se justifica pelo fato de o OSINT ter sido usado no país para fins militares, porém, antes disso, já se praticava a coleta de informações em fontes abertas. Por se tratar de uma cultura já deferida no país, é normal que os Estados Unidos olhem com mais atenção para a área de OSINT. Além disso, suas publicações variam sobre diversos temas, desde segurança da informação e mídias sociais até combate ao terrorismo.

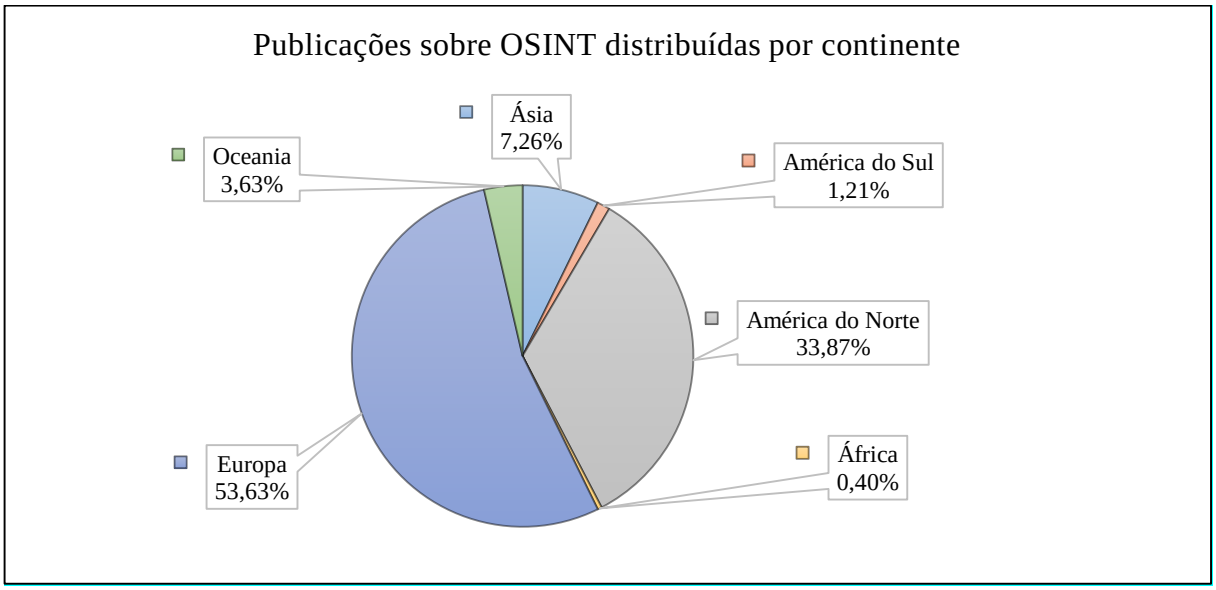
Nas publicações encontradas sobre OSINT no Reino Unido, observa-se uma preocupação em discutir formas de combater o terrorismo e os ciberataques, mas com foco na melhora da comunicação das forças a serviço do governo, como polícia e exército.

Quanto à China, as publicações encontradas envolvem a automatização de OSINT com plataformas, *frameworks*, ferramentas ou diferentes abordagens. A tendência de suas publicações segue para a execução de OSINT apoiada por técnicas de aprendizado de máquina para mineração de texto, tarefas de classificação e reconhecimento de padrões na área de segurança da informação e combate ao terrorismo.

As publicações da Itália seguem o mesmo caminho da China, combinando algoritmos de aprendizado de máquina para melhorar o desempenho de OSINT. A diferença é que suas publicações estão focadas em mídias sociais e big data, além de ontologia e semântica. Por fim, a Holanda tem, em sua grande maioria, publicações aplicando o OSINT para investigar e monitorar a atividade de sua população na internet.

Na Figura 19, demonstram-se as porcentagens sobre as publicações de OSINT distribuídas por continente.

Figura 19 – Publicações sobre OSINT distribuídas por continente



Fonte: o autor (2020).

Embora o país com maior concentração de publicações no OSINT seja os EUA, o continente com mais publicações é o continente europeu, com 53,63%, totalizando 133 publicações.

Muitos dos países do continente europeu que possuem estudos com OSINT abordam questões como segurança da informação e combate ao terrorismo, o que indica sua preocupação com tais questões e sua disposição em investir nesses temas.

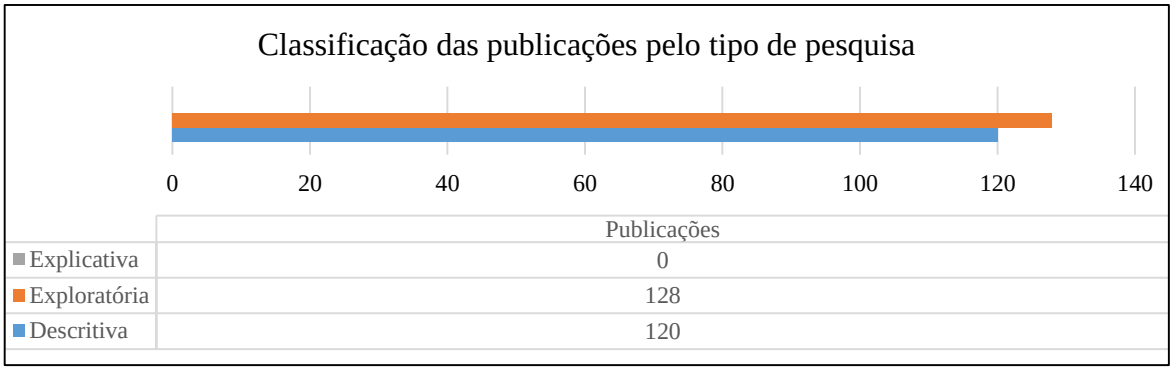
Os outros continentes, América do Sul, Africa, Oceania e Ásia, representam apenas 12,5% das publicações sobre OSINT encontradas na revisão sistemática da literatura, enquanto a América do Norte e a Europa representam juntas um total de 87,5%.

Foi realizada também a classificação das publicações por tipo de pesquisa: descritiva, exploratória ou explicativa. Para isso, foram verificados o resumo, a metodologia e os resultados das publicações.

As publicações que apresentaram resultados quantitativos, ou que abordavam a aplicação do OSINT em um determinado cenário, foram classificadas como descritivas. Já as publicações que abordaram uma discussão com hipóteses e análises qualitativas foram classificadas como exploratórias. Por fim, as pesquisas que tinham como foco explicar como o OSINT foi aplicado, bem como quais eram suas implicações, foram classificadas como explicativas.

Na figura 20, mostra-se a classificação das publicações por tipo de pesquisa.

Figura 20 – Classificação das publicações por tipo de pesquisa



Fonte: o autor (2020).

Feita a classificação, percebeu-se certo equilíbrio entre as publicações que adotaram tanto a pesquisa descritiva quanto a exploratória. No caso da pesquisa exploratória, o total foi de 128 publicações, enquanto que na pesquisa descritiva, o total foi de 120 publicações. Nenhuma publicação adotou a abordagem explicativa.

As publicações que adotaram a pesquisa descritiva lidaram com ferramentas, aplicações, abordagens e técnicas com as informações descobertas por OSINT. No caso das publicações que adotaram a pesquisa exploratória, discute-se o impacto que o OSINT pode causar quando aplicado em um determinado cenário e sob certas circunstâncias, como, por exemplo, para monitorar as atividades de uma determinada pessoa em uma rede social, estudo este que foi desenvolvido por Kanakaris, Tzovelekis e Bandekas (2018).

Quanto à pesquisa explicativa, não se encontrou nenhuma publicação que explicasse a necessidade de utilizar OSINT ou mesmo sua aplicação. Isto pode ser compreendido analisando as demais publicações encontradas na revisão sistemática da literatura, uma vez que elas abordam temas como: combate ao terrorismo, cybersegurança e questões de privacidade.

Etapa F (Download e Importação das Publicações para o Mendeley): Efetuou-se o download das publicações selecionadas na etapa D. Em seguida, todas as 248 publicações foram inseridas em um único diretório para serem importadas para o Mendeley.

Posteriomente, todas as publicações no Mendeley foram selecionadas e gerou-se um arquivo contendo todas as informações das publicações. O arquivo foi gerado com a extensão

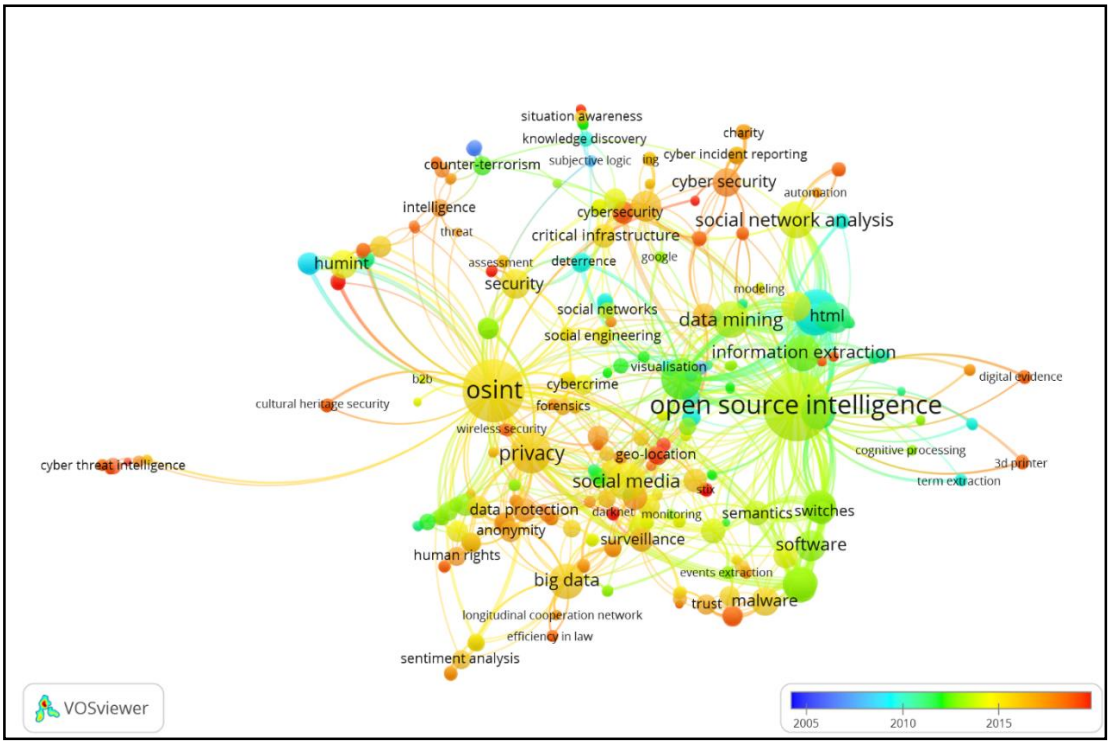
“.Ris” que, em sua composição, contém informações como título, local de publicação, autores, resumo, palavras-chave, número de páginas, ano de publicação, dentre outras informações.

Etapa G (Importação do .RIS para o VOSviewer e Análise dos Resultados): Exportou-se o arquivo .Ris com as informações das publicações inseridas no Mendeley (Etapa F) para utilizá-lo no VOSviewer. Com o VOSviewer, desenvolveu-se um mapa com base nos dados bibliográficos presentes no arquivo .Ris.

O mapa apresenta uma rede com as palavras-chave contidas nas 248 publicações encontradas na revisão sistemática da literatura. Conforme a quantidade de ocorrências das palavras-chave, ou seja, a quantidade de vezes em que elas aparecem nas publicações, molda-se um círculo ao seu redor.

Enquanto as palavras-chave com a maior quantidade de ocorrências estão dispostas em círculos maiores no mapa, as palavras-chave com menos ocorrências aparecem em círculos menores. Além disso, as palavras-chave estão relacionadas com outras palavras-chave. No mapa, cada palavra-chave está sendo representada por uma cor, e cada cor representa o ano em que a palavra-chave teve maior quantidade de menções. O mapa de palavras-chave é apresentado na Figura 21.

Figura 21 – Mapa de palavras-chave



Fonte: o autor (2020).

Analisando a Figura 21, percebe-se que as palavras-chave mais mencionadas nas publicações são: “OSINT” e “Open Source Intelligence”. Enquanto a palavra-chave “Open Source Intelligence” possui uma concentração no período entre 2012 e 2014, a palavra-chave “OSINT” tem maior concentração entre 2014 e 2016. Ambas estão posicionadas no centro e relacionam-se com as demais palavras-chave presentes na Figura 21.

A análise dos resultados obtidos na Figura 21 é apresentada na Tabela 16.

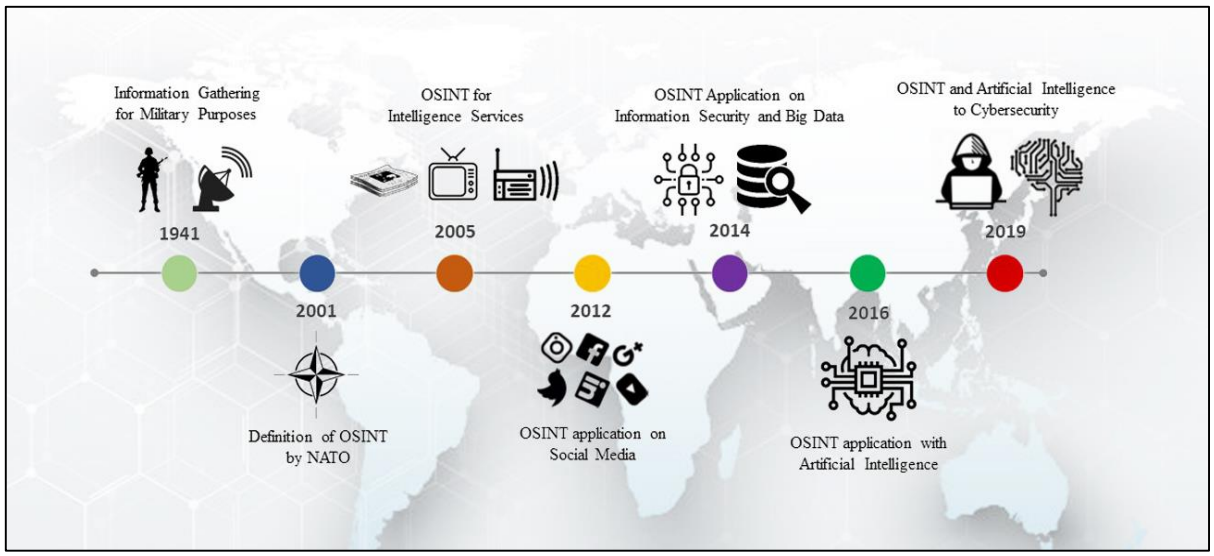
Tabela 16 – Análise dos resultados obtidos no mapa de palavras-chave

Período	**Palavras-chave mais mencionadas**	**Temas abordados**
Até 2010	HTML, Descoberta de Conhecimento, Redes Sociais e Terrorismo	- Extração de conhecimento em páginas web para combate e detecção de terrorismo
2012 a 2014	Redes Sociais, Mídias Sociais, Mineração de Dados, Semântica e Terrorismo	- Análise de informações de mídias e redes sociais - Mineração de dados para extrair conhecimento de textos
2014 a 2016	Cibersegurança, Big Data, Análise de Sentimentos, Privacidade, Malware, Engenharia Social e Direitos Humanos	- Aplicação de OSINT em Big Data - OSINT com Aprendizagem de Máquina e Processamento de Linguagem Natural - OSINT em Segurança da Informação
2016 a 2020	Cibersegurança, Automatização, Terrorismo, Evidências Digitais, CTI, Darknet, Anonimato e Legislação	- OSINT com Aprendizagem de Máquina e Processamento de Linguagem Natural - OSINT em Segurança da Informação e Forense Digital - OSINT para apoiar o desenvolvimento de leis do âmbito digital

Fonte: o autor (2020).

Com base na Figura 21 e na Tabela 16, desenvolveu-se uma linha do tempo, descrevendo a evolução e as principais áreas de aplicação de OSINT, conforme pode-se observar na Figura 22.

Figura 22 – Linha do tempo sobre a evolução de OSINT



Fonte: o autor (2020).

Em 1941, já se praticava a coleta de informações em fontes abertas. As publicações identificadas neste trabalho, como a de Clarke (2015), descrevem o monitoramento, por parte do governo americano, das transmissões de rádio entre Alemanha e Japão, durante a segunda guerra mundial. No entanto, somente em 2001 o OSINT foi definido pela Organização do Tratado do Atlântico Norte (OTAN), passando a ser usado em serviços de inteligência.

As publicações datadas de 2005 descrevem a expansão de OSINT, passando a coletar informações de jornais, televisão e rádio. Em 2012, com o advento das mídias sociais, o OSINT passou a ser utilizado para buscar informações de diversos tipos, como textos, fotos e vídeos. Em 2014, o volume de informações compartilhadas na internet cresceu exponencialmente, muito em função da quantidade de novos usuários em mídias sociais. Com isso, surgiram tecnologias para lidar com o armazenamento e manuseio de informações, como o Big Data.

Ainda em 2014, OSINT começou a ser abordado como uma possível solução para encontrar informações pertinentes na internet, e também passou a ser aplicado na área de segurança da informação, apoiando organizações e governo regionais, e abordando, principalmente, temas como privacidade e anonimato.

Em 2016, o OSINT passou a ser aplicado em conjunto com a IA. Apesar de o OSINT encontrar informações na internet, foi necessário utilizar técnicas de IA para extrair o conhecimento delas.

Em marketing digital, o OSINT passou a ser utilizado com a análise de sentimentos e as técnicas de mineração de texto, já para a área de segurança da informação, com técnicas de classificação e geração de regras.

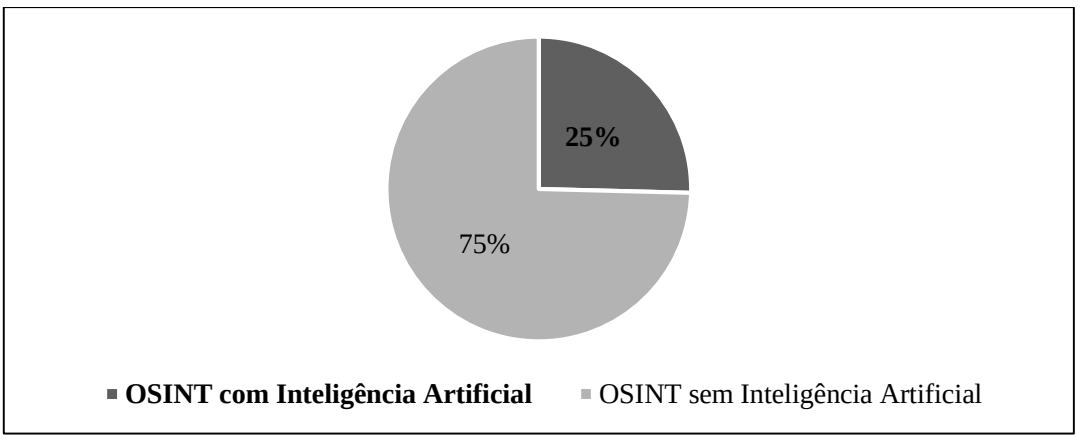
Em 2019, intensificou-se a aplicação de OSINT em conjunto com IA. Nesse período, a maior concentração das publicações foi para a área de segurança da informação, principalmente na busca por vulnerabilidades em sistemas de informação, no desenvolvimento de leis para o âmbito digital e para apoiar o campo de forense digital.

Percebendo essa tendência de se utilizar o OSINT em conjunto com IA, a partir de 2014, realizou-se uma busca nas publicações selecionadas na revisão sistemática da literatura para descobrir quais abordaram a inteligência artificial. Para isso, todas as 248 publicações foram importadas para o Mendeley, e efetuou-se uma busca com os seguintes temas: “Artificial Intelligence” OR “Natural Language Processing” OR “Pattern Recognition” OR “Robotic” OR “Machine Learning”.

A pesquisa foi feita usando os temas em inglês porque, dentre as publicações identificadas na revisão sistemática da literatura, todas estão nesse idioma. Definiram-se esses temas com base no estudo de Vijayakumar e Sheshadri (2019), referenciado neste trabalho, sobre aplicações das subáreas da IA em bibliotecas virtuais.

Foram encontradas 63 publicações sobre OSINT abordando IA, seja na sua execução ou na análise dos resultados. Esse número representa 25% de todas as publicações identificadas na revisão sistemática da literatura, como pode-se observar na Figura 23 (No apêndice B deste trabalho são descritas as 63 publicações).

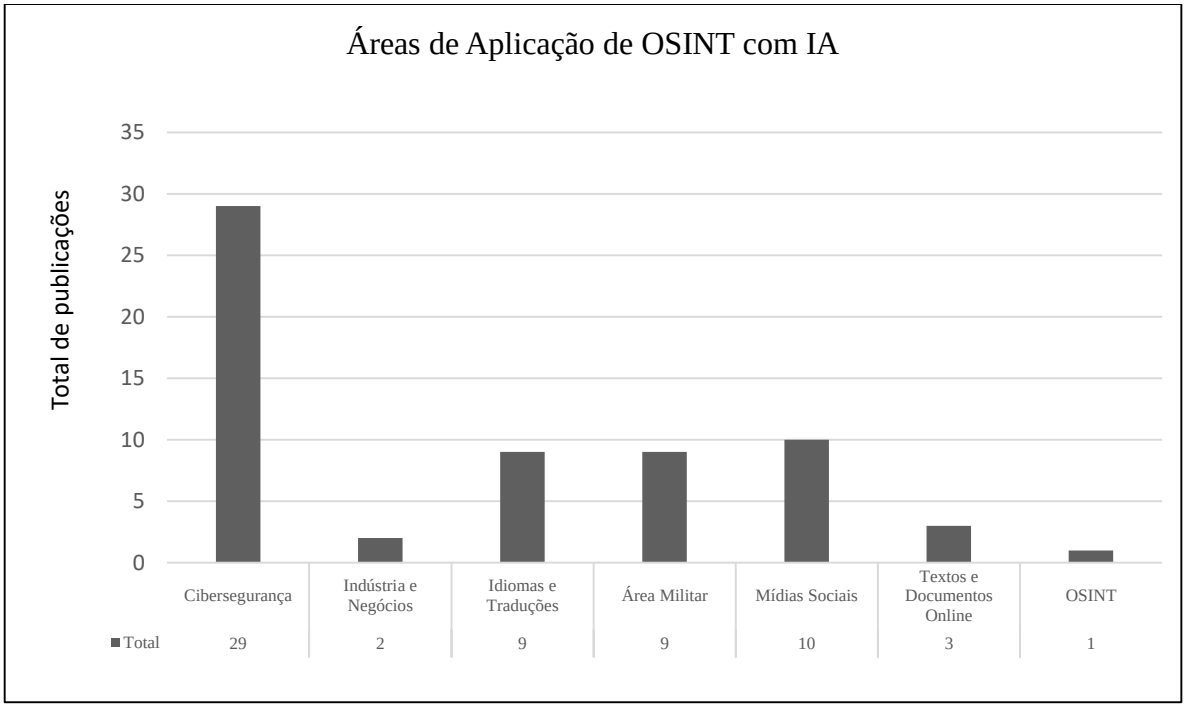
Figura 23 – Publicações que abordam OSINT com IA



Fonte: o autor (2020).

Em adição, foi feita uma análise sobre as áreas em que o OSINT está sendo aplicado em conjunto com IA. Na figura 24, apresentam-se os resultados.

Figura 24 – Áreas de aplicação de OSINT com IA



Fonte: o autor (2020).

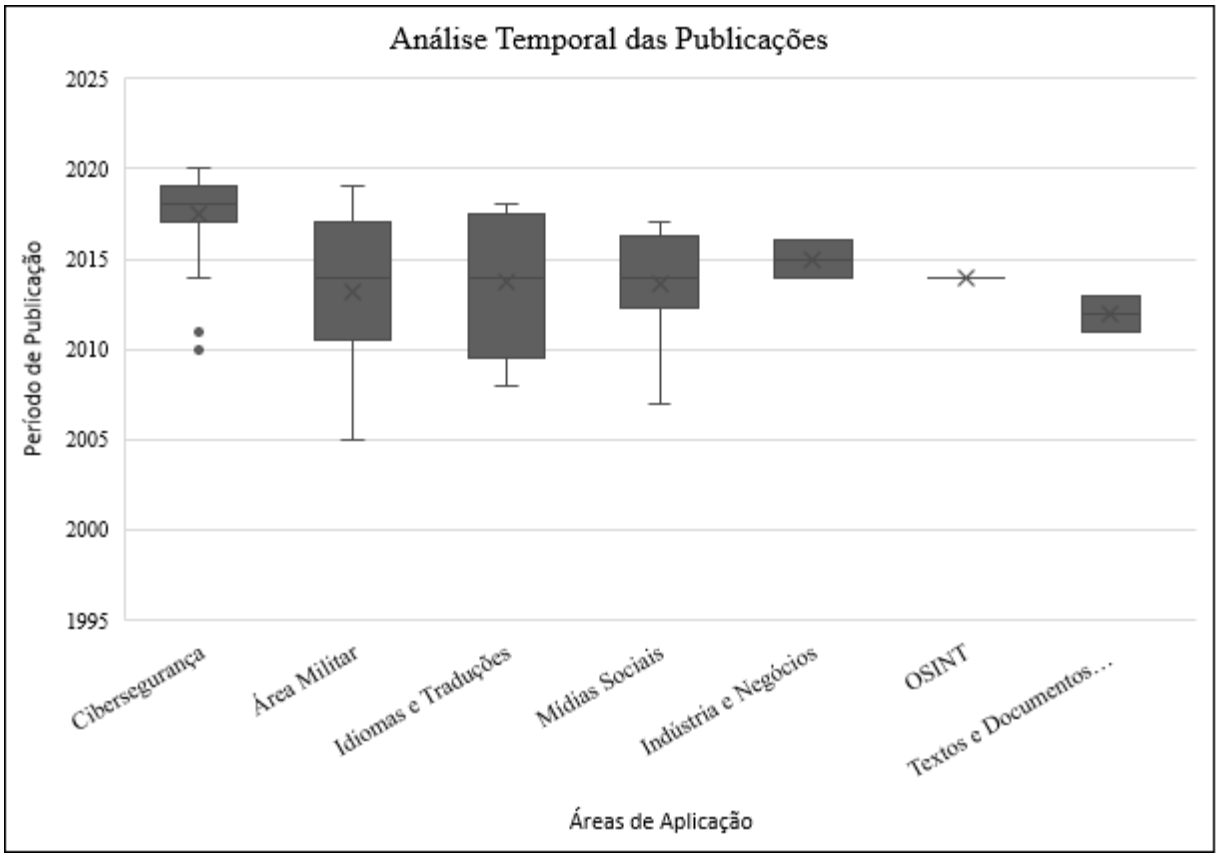
Analisando a Figura 24, é possível identificar quais são as principais áreas em que o OSINT está sendo aplicado com a IA. Das 63 publicações, apenas uma é específica para a execução do OSINT, sem abordar outra área de forma objetiva.

A maior concentração está na área de cibersegurança, com um total de 29 publicações. A área de mídias sociais aparece em segundo, com 10 publicações. Já a área de idiomas e traduções aparece em terceiro lugar, com 9 publicações. Aqui, estão as publicações para tradução de textos e falas em outros idiomas, como o mandarim por exemplo.

Em terceiro lugar, aparece também a área militar, com 9 publicações. As áreas de indústria e negócios e de textos e documentos on-line aparecem com as menores concentrações, 2 e 3 publicações, respectivamente. Por fim, consta apenas 1 publicação sobre OSINT, focada especificamente em sua execução, e sem abordar outra área.

Feita a análise das áreas de aplicação de OSINT com IA, realizou-se a análise temporal para descobrir como as publicações que abordam o assunto estão distribuídas. Tal análise é mostrada na Figura 25.

Figura 25 – Análise temporal das publicações que abordam OSINT com IA



Fonte: o autor (2020).

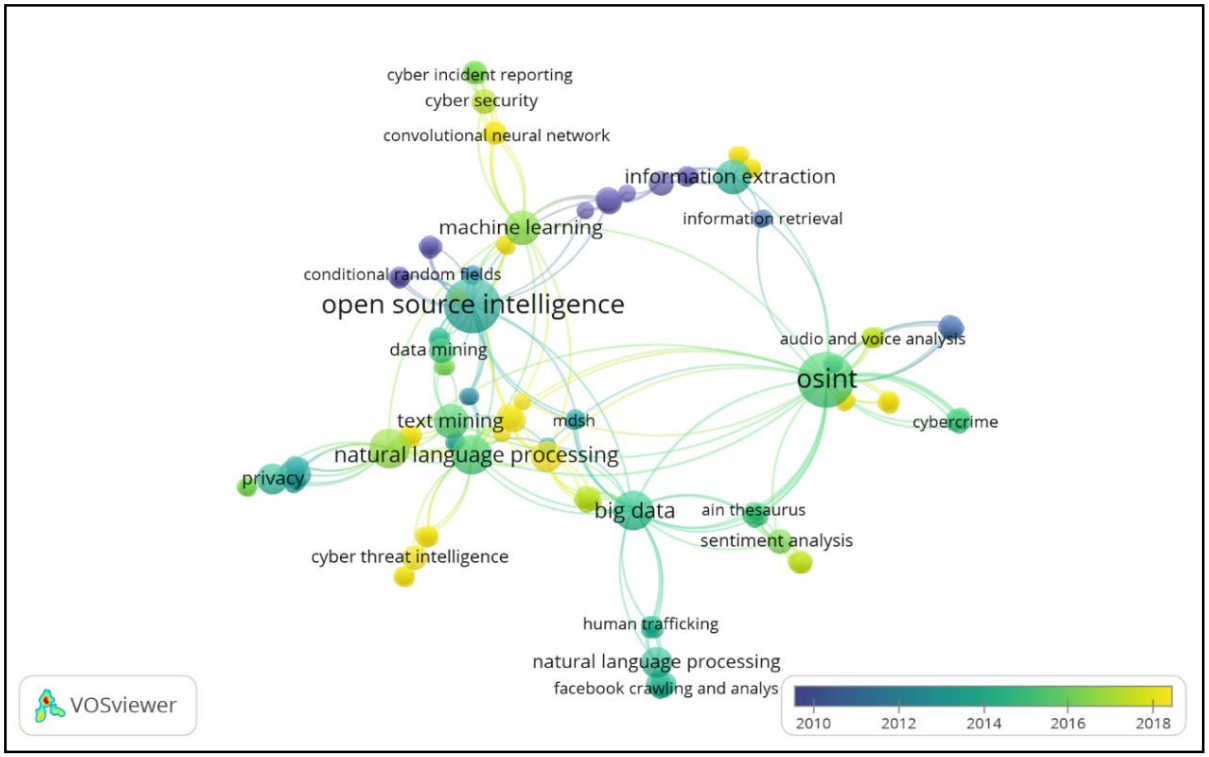
Analisando a Figura 25, percebe-se quais são os anos de maior concentração de publicações para cada área. No período de 2005 a 2010, as publicações apenas recomendam a IA como uma possível solução, enquanto os trabalhos posteriores mostram efetivamente sua aplicação com OSINT.

As áreas “textos e documentos on-line”, “OSINT” e “negócios e indústria” possuem publicações até 2016. Já as áreas “mídias sociais” e “idiomas e traduções” avançam até 2017. A área militar é a que possui um período maior de publicações, de 2005 a 2019. Por fim, a área da cibersegurança, que possui a maior quantidade de publicações, tem sua concentração entre 2017 a 2019.

Considerando a tendência de publicações de OSINT com IA apresentada nesta revisão sistemática da literatura, especificamente nas Figuras 22, 23, 24 e 25, e nas Tabelas 15 e 16, foram realizadas novamente as Etapas F e G com as 63 publicações que utilizam OSINT com IA.

Diferentemente da Figura 21, que apresenta o mapa com todas as palavras-chave presentes nas publicações encontradas na revisão sistemática da literatura, na Figura 26, apresenta-se o mapa das palavras-chave somente em publicações que abordam a utilização de OSINT com IA.

Figura 26 – Mapa de palavras-chave das publicações que abordam OSINT com IA



Fonte: o autor (2020).

A maior concentração das publicações que abordam OSINT com IA aparece entre 2010 e 2018. Na maioria das vezes, essas publicações apontam questões de segurança da informação e mídias sociais. Isso indica que a IA passou a ser estudada como uma possível solução para apoiar a execução do OSINT.

Antes de 2010, as publicações apenas recomendavam a IA. Nessa época, a internet estava começando a evoluir e era necessário procurar formas de coletar e analisar informações não-estruturadas e distribuídas em diferentes países, cada um com suas perspectivas e necessidades.

As publicações sobre OSINT com IA, datadas entre 2010 e 2014, apresentam em sua maioria o uso de PLN para extrair conhecimento de mídias sociais e tecnologias. Nesse período, começam a surgir assuntos como privacidade, tráfico de pessoas e crimes cibernéticos.

As publicações datadas entre 2014 e 2019 abordam não apenas o PLN, como também a aprendizagem de máquina, para realização de análise de sentimentos, mineração de texto e análise de arquivos de mídia, como uma solução para problemas de cibersegurança e como suporte para tecnologias como o Big Data.

3.4 CONDUÇÃO DOS EXPERIMENTOS COMPUTACIONAIS

Realizou-se este trabalho tendo como inspiração os estudos de Rico *et al*. (2018), que desenvolveram uma abordagem de OSINT para apoiar operações de segurança cibernética, de Lee e Shon (2016), que apresentaram uma abordagem de OSINT para inspecionar sistemas de controle de infraestruturas críticas, de Li *et al*. (2018), que desenvolveram uma abordagem de OSINT para obter informações sobre ameaças CTI, e de Zhao, Cao e Liu (2015), que utilizaram uma estrutura básica para o desenvolvimento de abordagens, ferramentas ou *frameworks* OSINT.

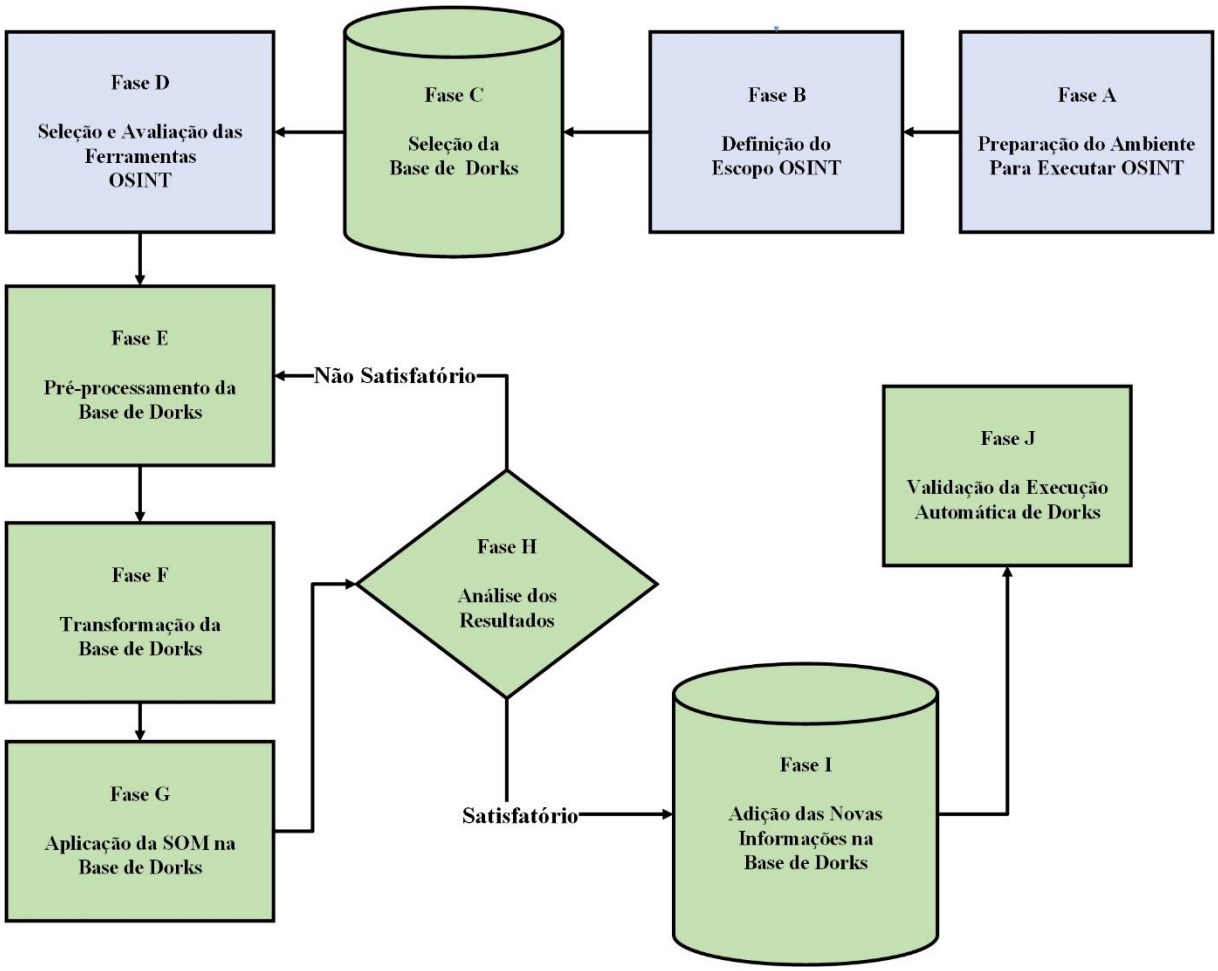
Os quatro estudos constam na Fundamentação Teórica deste trabalho.

A abordagem proposta neste trabalho se difere das demais citadas por ser específica para a prática do Google Hacking e por ter o objetivo de executar Dorks de forma automática. Sendo assim, utilizou-se aqui uma base de Dorks, o GHDB, descrita na Seção 3.2 deste capítulo. Além disto, a abordagem aqui proposta também utilizou técnicas de IA para extrair conhecimento do GHDB e, assim, torná-la possível de ser executada de forma automática.

Quanto às contribuições dos quatro estudos para desenvolvimento deste trabalho, pode-se dizer que as abordagens propostas por Rico *et al*. (2018), Lee e Shon (2016) e Li *et al*. (2018), que descrevem a preparação para executar a abordagem, a seleção das fontes abertas onde a abordagem será executada e a definição das ferramentas OSINT em que serão executadas, foram incorporadas na abordagem aqui proposta. Quanto ao estudo de Zhao, Cao e Liu (2015), as partes que abordam a definição do sistema operacional e a utilização de técnicas para análise computacional também foram incorporadas na abordagem proposta no presente trabalho.

A abordagem aqui proposta foi dividida em 10 fases para executar Dorks de forma automática. Tais fases podem ser observadas a seguir na Figura 27.

Figura 27 – Fases da abordagem de OSINT com IA para execução automática de Dorks



Fonte: o autor (2020).

Como pode ser visualizado na Figura 27, a divisão das fases em duas cores, verde e azul, revela que as fases na cor azul estão presentes nas abordagens selecionadas, e que fases na cor verde são propostas na abordagem deste trabalho.

Em relação à execução da abordagem, dividiram-se as fases em três grupos. O primeiro grupo contendo as fases A, B, C e D, o segundo grupo contendo as fases E, F, G, H e I e o terceiro grupo contendo a fase J.

Descrevem-se, na Tabela 17, os três grupos da abordagem proposta neste trabalho.

Tabela 17 – Grupos da abordagem proposta neste trabalho

Grupo	Fases
1º	A – Preparação do Ambiente para Executar OSINT B – Definição do Escopo OSINT C – Seleção da Base de Dorks D – Seleção e Avaliação das Ferramentas OSINT
2º	E – Pré-processamento da Base de Dorks F – Transformação da Base de Dorks G – Aplicação da rede SOM na Base de Dorks H – Análise dos Resultados I – Adição das Novas Informações na Base de Dorks
3º	J – Validação da Execução Automática de Dorks

Fonte: o autor (2020).

As primeiras 4 fases (A-D) representam o primeiro grupo da abordagem. Esse grupo envolve a preparação para a execução de OSINT e a definição dos recursos disponíveis; dentre eles, a base de Dorks. O segundo grupo (E-I) é composto pelas fases em que se aplicam o PLN e a rede SOM na base de Dorks, em que se analisam os resultados obtidos e em que se adicionam as novas informações descobertas pela aplicação da SOM na base de Dorks.

O terceiro grupo, que contém a fase J, trata-se da validação da execução automática de Dorks com as ferramentas OSINT, a base de Dorks definida no primeiro grupo e as técnicas de IA e PLN aplicadas na base do grupo 2. Nessa fase, executam-se as Dorks de forma manual e automática e comparam-se os resultados, avaliando assim o desempenho da abordagem proposta neste trabalho.

Descrevem-se, a seguir, as 10 fases da abordagem de OSINT com IA para execução automática de Dorks, conforme demonstrado anteriormente na Figura 27 e Tabela 17.

Fase A – Preparação do Ambiente para Executar OSINT

Nesta fase, definiu-se o ambiente onde OSINT será executado. Para isso, foram selecionados os sistemas operacionais em que serão executadas as ferramentas OSINT, tanto de forma anônima quanto de forma não-anônima, bem como um software para a criação das máquinas virtuais.

Fase B – Definição do Escopo de OSINT

Nesta fase, definiu-se o escopo de OSINT, ou seja, qual o objetivo para utilizar OSINT e quais práticas serão aplicadas. Além disso, determinou-se qual é o alvo da prática de OSINT.

Fase C – Seleção da Base de Dorks

Nesta fase, selecionou-se a base de Dorks para apoiar a execução automática do Google Hacking. Definiu-se a base a partir da literatura pesquisada e referenciada neste trabalho.

Fase D – Seleção e Avaliação das Ferramentas OSINT

Nesta fase, selecionaram-se ferramentas para executar a prática OSINT definida na fase B. Para isso, foi feita uma pesquisa sobre ferramentas na literatura e na internet, utilizando o método apresentado no estudo de Lee e Shon (2016).

Com as ferramentas selecionadas, todas foram instaladas em seus respectivos sistemas operacionais (Linux ou Windows). Terminada a instalação, alguns critérios de avaliação foram desenvolvidos, a fim de se avaliar as ferramentas e definir qual será utilizada nesta abordagem.

Fase E – Pré-processamento da Base de Dorks

Nesta fase, efetuou-se o pré-processamento da base de Dorks definida na fase C. Para isso, foram desconsiderados e removidos os atributos da base, assim como os parâmetros das Dorks, a fim de torná-las capazes de serem executadas em qualquer página web, e criou-se uma legenda numérica para cada categoria. Nesta fase, também foi feito a identificação e remoção de *outliers* da base de Dorks, além da remoção de *stopwords* utilizando PLN.

Fase F – Transformação da Base de Dorks

Nesta fase, converteu-se a Dork de valor textual em valor numérico em ASCII. Em seguida, criaram-se os atributos para inserir a Dork com o valor numérico em sua base. Tanto a conversão da Dork quanto a inserção na base são efetuadas por um código em Python.

Fase G – Aplicação da Rede SOM na Base de Dorks

Nesta fase, aplicou-se a rede SOM na base de Dorks pré-processada e transformada. Para isso, utilizou-se o software Viscovery SOMine. A aplicação da SOM teve por objetivo extrair o conhecimento da base de Dorks.

O conhecimento será produzido na análise dos mapas gerados pela aplicação da rede SOM na base de Dorks. Essa análise que será realizada nos mapas na fase H irá gerar uma nova classificação para grupos de Dorks, tornando-as aptas para serem executadas de forma automática.

Fase H – Análise dos Resultados

Nesta fase, analisaram-se os resultados obtidos pela aplicação da rede SOM na base de Dorks na fase G. Aqui, são discutidos os resultados obtidos pela SOM e apresentados os valores das métricas de desempenho do mapa.

As métricas utilizadas para avaliar os mapas gerados pela SOM são o Erro Topográfico (ET) e o Erro de Quantização (EQ), ambos descritos anteriormente na Tabela 9 e apresentados no Capítulo 2.6.2 deste trabalho.

Fase I – Adição das Novas Informações na Base de Dorks

Nesta fase, adicionou-se o conhecimento obtido dos resultados da aplicação da SOM, descritos na fase H, na base de Dorks, definida na fase C. Para isso, realizou-se uma nova classificação, baseando-se nos agrupamentos gerados pela rede SOM.

Fase J – Validação da Execução Automática de Dorks

Nesta fase, foram executadas as Dorks com as novas informações descobertas com a SOM na fase I e com a execução automática de Dorks com a ferramenta definida na fase C. Também se executaram as Dorks de forma manual e foram comparados os resultados com a execução automática.

No próximo capítulo, os resultados obtidos dos experimentos computacionais são apresentados e discutidos.

4 APRESENTAÇÃO E DISCUSSÃO DOS RESULTADOS

Neste capítulo, são apresentados e discutidos os resultados dos experimentos computacionais realizados para a seleção e avaliação das ferramentas OSINT, o pré-processamento e transformação da base de Dorks, a aplicação da rede SOM para extração de conhecimento da base de Dorks, a análise dos resultados obtidos por meio da rede SOM e a validação da abordagem comparando a execução automática de Dorks com a execução manual.

A abordagem proposta de OSINT com IA para execução automática de Dorks foi dividida em 3 grupos distintos:

- Experimentos computacionais do primeiro grupo da abordagem, que reúne as fases A, B, C e D.

- Experimentos computacionais do segundo grupo da abordagem, que reúne as fases E, F, G, H e I;

- Experimentos computacionais do terceiro grupo da abordagem, que reúne a fase J.

4.1 EXPERIMENTOS COMPUTACIONAIS DO PRIMEIRO GRUPO DA ABORDAGEM

A seguir, apresentam-se os resultados dos experimentos computacionais do primeiro grupo da abordagem, composto pelas fases:

FASE A – PREPARAÇÃO DO AMBIENTE PARA EXECUTAR OSINT

Escolheu-se o sistema operacional **Windows10 Home Edition** para testar as ferramentas OSINT, descobertas na fase C, para arquiteturas Windows, e o sistema operacional **ParrotSec** para testar as ferramentas OSINT, para arquiteturas Linux. Escolheu-se o sistema operacional **ParrotSec** pela possibilidade de testar a execução das ferramentas OSINT das formas “Anônima” e “Não-Anônima”. Por fim, escolheu-se o software **VirtualBox** para criação da máquina virtual com os sistemas operacionais definidos anteriormente.

FASE B – DEFINIÇÃO DO ESCOPO DE OSINT

Definiu-se o escopo do OSINT, ou seja, qual o objetivo de utilizar OSINT e quais práticas serão utilizadas. Escolheu-se a prática do Google Hacking, pois, segundo Bae, Lim e Cho

(2016) e Mider, Garlicki e Jan (2019), é uma das principais práticas para OSINT na fase inicial de um *Pentest*.

Quanto ao alvo da prática do Google Hacking, não foi necessário determinar uma página web específica para sua execução. É possível executar a Dork sem o parâmetro "Site" e obter como resultado todas as páginas em que o Google encontrar vulnerabilidades, como demonstra a classificação das palavras que compõe a Dork feita por Pan *et al*. (2012).

FASE C – SELEÇÃO DA BASE DE DORKS

Nesta fase, selecionou-se a base de Dorks para apoiar a execução automática do Google Hacking. Escolheu-se a base Google Hacking Database da organização *Offensive Security* por se tratar da maior e mais representativa base de Dorks on-line, segundo Zhang, Notani e Gu (2015) e Meucci e Muller (2014).

FASE D – SELEÇÃO E AVALIAÇÃO DAS FERRAMENTAS OSINT

Nesta fase, foi feita uma busca pelas ferramentas necessárias para executar Dorks de forma automática. Encontrou-se um total de 15 ferramentas, mas somente uma foi citada na literatura, a "SiteDigger3", descrita pelos autores Bae, Lim e Cho (2016).

Para encontrar as outras 14 ferramentas, foram utilizados dois mecanismos de busca e um *framework* OSINT, tal como proposto no trabalho de Lee e Shon (2016). Os Mecanismos de busca utilizados foram: **Carrot2** e **Google**. Já o *framework* OSINT utilizado foi o **OsintFramework.**

Em ambos os mecanismos, buscaram-se pelas ferramentas utilizando as palavras-chave "Dork" e "Automat", de modo a encontrar ferramentas que, em sua descrição, possuíssem a palavra "Dork".

Após identificadas, foram instaladas as ferramentas para sistema operacional Linux no **ParrotSec**, e as ferramentas para sistema operacional Windows, no **Windows10 Home Edition**. Em seguida**,** baseando-se nas recomendações de Bae, Lim e Cho (2016) para trabalhos futuros, definiram-se os seguintes critérios para selecionar a ferramenta para execução automática de Dorks. Tais critérios são descritos na Tabela 18.

Tabela 18 – Critérios de seleção das ferramentas para execução automática de Dorks

ID	Critério	Resposta
01	A ferramenta permite a inserção de Dorks?	Sim / Não
02	Qual a quantidade de Dorks que a ferramenta consegue executar?	Valor Numérico
03	Qual a quantidade de execuções até que o Google faça o bloqueio do tráfego?	Valor Numérico
04	Qual a quantidade de vulnerabilidades encontradas?	Valor Numérico
05	Qual o tempo de execução de cada ferramenta?	Valor Numérico

Fonte: Autor (2020).

Dos critérios definidos e apresentados na Tabela 18, apenas o terceiro não é baseado no estudo de Bae, Lim e Cho (2016). Incluiu-se esse critério porque o Google possui um mecanismo de defesa em seu site para identificar constantes buscas, que são realizadas utilizando *strings* consideradas "maliciosas" pelo Google. Descreve-se a seguir a aplicação dos cinco critérios de seleção das ferramentas para execução automática de Dorks.

a) Primeiro Critério de Avaliação: No primeiro critério, apresentado na Tabela 19, verifica-se se a ferramenta permite a inclusão de Dorks. Aqui, avalia-se a possibilidade de utilizar as Dorks da base definida na fase C **(Seleção da Base de Dorks)** desta abordagem.

Tabela 19 – Avaliação das ferramentas de acordo com o primeiro critério

Ferramenta	Sistema operacional	URL	Permite a inclusão de Dorks
SiteDigger3	Windows	http://downloadcenter.mcafee.com/products/tools/foundstone/sitedigger3.msi	Não
Pagodo	Linux	https://github.com/opsdisk/pagodo	Sim
ATScan	Linux	https://github.com/AlisamTechnology/ATSCAN	Não
Doork	Linux	https://github.com/AeonDave/doork	Sim
DorkMe	Linux	https://github.com/blueudp/DorkMe	Sim
UltimateDork	Linux	https://github.com/jaxBCD/Ultimate-Dork	Não
DorkScan	Linux	https://github.com/SivertPL/Dorkscan-Project	Não
XgDork	Linux	https://github.com/E4rr0r4/XGDork	Não
DorkNet	Linux	https://github.com/NullArray/DorkNet	Sim
GoogleDocker	Linux	https://github.com/nerrorsec/GoogleDorker	Não
Fast-Recon	Linux	https://github.com/DanMcInerney/fast-recon	Não
VTI-Dork	Linux	https://github.com/Neo23x0/vti-dorks	Não
Dork-CLI	Linux	https://github.com/jgor/dork-cli	Não
Dorks	Linux	https://github.com/USSCltd/dorks	Sim
Bingoo	Linux	https://github.com/Hood3dRob1n/BinGoo	Não

Fonte: o autor (2020).

Analisando a Tabela 19, observa-se que apenas 5 das 15 ferramentas encontradas permitem a inserção de Dorks. As outras 10 também executam Dorks de forma automática, mas não permitem a inserção de novas Dorks na base definida na fase C.

Em relação às ferramentas (5) que permitem a inserção de Dorks, é necessário que as Dorks estejam separadas em um arquivo de texto com a extensão ".txt". Assim, baseando-se nos resultados do primeiro critério de seleção das ferramentas, 5 foram selecionadas, e 10, desconsideradas.

b) Segundo Critério de Avaliação: Este critério verifica qual a quantidade de Dorks que a ferramenta consegue executar. Por isso, selecionou-se uma amostra de Dorks a ser executada. Foram escolhidas como amostra todas as Dorks da categoria "Advisories and Vulnerabilities", totalizando 1.979 Dorks, uma vez que esta possui a maior quantidade de Dorks na base.

Em seguida, com as 1.979 Dorks selecionadas como amostra, foram criados 9 arquivos de textos com quantidades diferentes de Dorks para avaliar a capacidade de execução de cada ferramenta. Desta forma, cada ferramenta foi executada 7 vezes, cada execução com uma quantidade específica de Dorks. Neste critério de avaliação, foram realizadas 35 execuções no total.

Cada execução, feita para avaliar o segundo critério de seleção de ferramentas, foi realizada em um dia, ou seja, essa avaliação durou 35 dias. O procedimento foi feito dessa forma para garantir que o endereço IP da máquina que realizou os experimentos se renovasse. Como resultado, não houve bloqueio do Google em nenhuma das execuções.

Na Tabela 20, apresenta-se a avaliação das cinco ferramentas de acordo com o segundo critério, ou seja, a quantidade de Dorks que a ferramenta consegue executar.

Tabela 20 – Avaliação das ferramentas de acordo com o segundo critério

Ferramenta	**Quantidade de Dorks**						
	50	**100**	**225**	**500**	**1000**	**1250**	**1500**
Pagodo	Sim	Sim	Sim	Sim	Não	Não	Não
Doork	Sim	Sim	Sim	Sim	Sim	Sim	Não
DorkMe	Sim	Sim	Sim	Sim	Sim	Sim	Não
DorkNet	Sim	Sim	Sim	Sim	Sim	Não	Não
Dorks	Sim	Sim	Sim	Sim	Não	Não	Não

Fonte: o autor (2020).

Das cinco ferramentas selecionadas de acordo com o primeiro critério de avaliação, apenas “Doork” e “DorkMe” conseguiram executar a maior quantidade de Dorks no segundo critério de avaliação. As duas ferramentas executaram até o arquivo .txt, com um total de 1.250 Dorks.

Já as outras três ferramentas tiveram resultados inferiores. As ferramentas “Pagodo” e Dorks” conseguiram executar somente até o arquivo .txt com o total de 500 Dorks. Já a ferramenta “DorkNet” conseguiu executar até o arquivo .txt com o total de 1.000 Dorks. Nesta segunda avaliação, nenhuma das cinco ferramentas conseguiram executar toda a amostra, totalizando 1.979 Dorks na categoria “Advisories and Vulnerabilities” da base de Dorks definida na fase C.

Desta forma, para que a base de Dorks possa ser executada de forma automática, ou seja, para que todas as 4.211 Dorks sejam classificadas em 14 categorias, conforme apresentado no Capítulo 3, Seção 3.2 deste trabalho, será necessário dividi-la em grupos menores de Dorks, de modo que todas possam ser executadas pela ferramenta selecionada nesta fase.

c) Terceiro Critério de Avaliação: Este critério verifica qual a quantidade de execuções que a ferramenta consegue realizar até que o Google faça o bloqueio do tráfego.

Aqui, avaliou-se quantas vezes as duas ferramentas selecionadas no segundo critério de avaliação conseguem ser executadas até o bloqueio de tráfego do Google. Esse bloqueio acontece quando o Google detecta que Dorks estão sendo realizadas.

Para a realização da avaliação do terceiro critério, o arquivo de texto com as 1.250 Dorks foi selecionado e tentou-se executá-lo repetidamente em cada ferramenta. A escolha desse arquivo de texto se deu por se tratar da capacidade máxima das duas ferramentas. Cada ferramenta foi executada em um dia diferente, para assim garantir a renovação do endereço IP. Na Tabela 21, apresenta-se a avaliação das ferramentas de acordo com o terceiro critério.

Tabela 21 – Avaliação das ferramentas de acordo com o terceiro critério

Ferramenta	**1ª Execução**	**2ª Execução**	**3ª Execução**
Doork	Conseguiu	Conseguiu	Não Conseguiu
DorkMe	Conseguiu	Conseguiu	Não Conseguiu

Fonte: o autor (2020).

Ambas as ferramentas indicadas na Tabela 21 conseguiram executar duas vezes a amostra de Dorks sem o bloqueio de tráfego do Google, mas, na terceira tentativa de cada ferramenta, houve o bloqueio. Sendo assim, as duas ferramentas foram avaliadas de acordo com o próximo critério: a quantidade de vulnerabilidades que elas conseguem encontrar.

d) Quarto Critério de Avaliação: Este critério de avaliação verifica qual a quantidade de vulnerabilidades que cada ferramenta consegue encontrar. Por isso, selecionou-se outra amostra de Dork para executar as ferramentas selecionadas até o terceiro critério de avaliação, ou seja, "Doork" e "DorkMe". Como no quarto critério será avaliada a quantidade de resultados que cada ferramenta irá trazer, ou seja, a quantidade de vulnerabilidades encontradas, selecionou-se uma única Dork. A saber: "**intitle:"index of" /content/admin/".**

Essa Dork foi escolhida por ser a mais recente na base no momento da execução do experimento, isto é, dezembro de 2019. Ambas as ferramentas ("Doork" e "DorkMe") encontraram um total de 654 sites com vulnerabilidades.

e) Quinto Critério de Avaliação: Este critério verifica qual o tempo de execução de cada ferramenta. Sendo assim, avaliou-se a duração da execução de cada uma das ferramentas que passaram pelos 4 critérios anteriores: "Doork" e "DorkMe".

Para fazer a avaliação de acordo com o quinto critério, foram selecionadas como amostra todas as Dorks da categoria "Sensitive Online Shopping Info", uma vez que essa categoria possui a menor quantidade de Dorks, 9 no total. Assim, criou-se um arquivo .txt com essas 9 Dorks. Em seguida, executou-se cada ferramenta três vezes com o mesmo arquivo de texto com as 9 Dorks. Como ambas as ferramentas exibem o tempo de execução, não foi necessário cronometrá-lo. Na Tabela 22, apresenta-se a avaliação das ferramentas de acordo com o quinto critério.

Tabela 22 – Avaliação das ferramentas de acordo com o quinto critério

Ferramenta	Quantidade de Dorks	Tempo (segundos)	Tempo Total
Doork	9	57 segundos por Dork	8 minutos e 55 segundos
DorkMe	9	49 segundos por Dork	7 minutos e 35 segundos

Fonte: o autor (2020).

A ferramenta “DorkMe” apresentou um melhor desempenho e conseguiu executar as Dorks mais rapidamente do que a ferramenta “Doork”, com uma diferença de 1 minuto e 20 segundos, em uma execução automática de 9 Dorks. Esse tempo crescerá consideravelmente conforme o aumento do número de Dorks. Assim, definiu-se por utilizar, neste trabalho, a ferramenta DorkMe para a execução automática de Dorks.

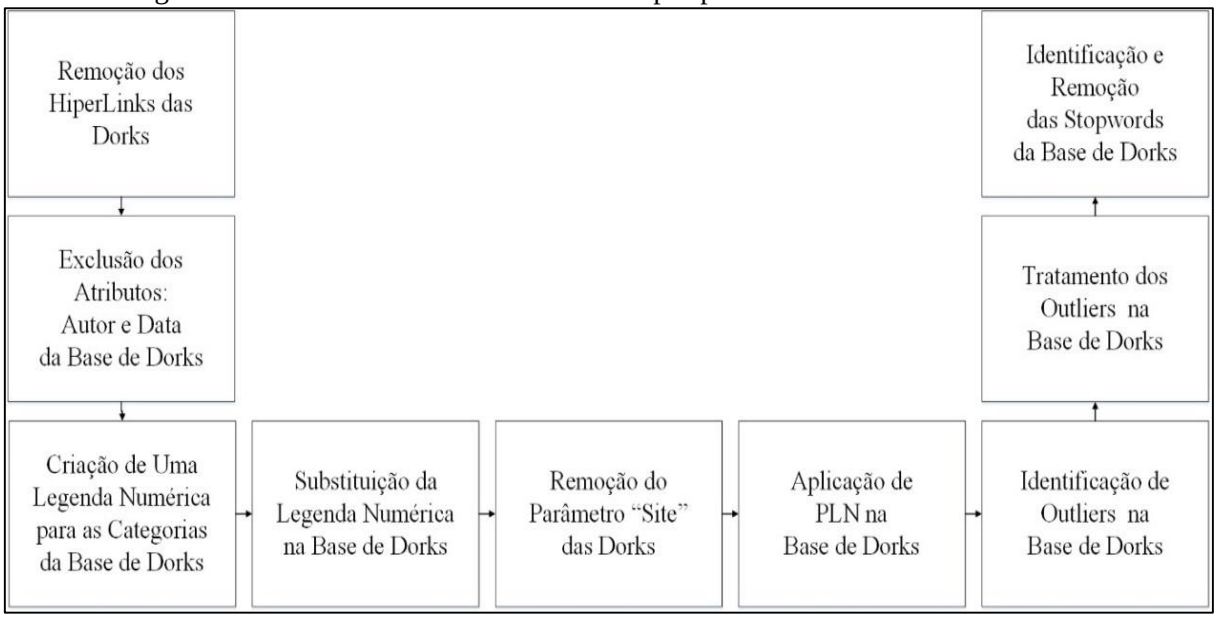
4.2 EXPERIMENTOS COMPUTACIONAIS DO SEGUNDO GRUPO DA ABORDAGEM

A seguir, apresentam-se os resultados dos experimentos computacionais do segundo grupo da abordagem, composto pelas fases:

FASE E – PRÉ-PROCESSAMENTO DA BASE DE DORKS

Nesta fase, foi pré-processada a base de Dorks definida na fase C **(Seleção da Base de Dorks)** deste estudo. Para isso, as Dorks disponíveis na base foram selecionadas e copiadas para uma planilha em Excel, que foi transformada, em seguida, em um arquivo .CSV. Este tipo de arquivo permite que a base seja importada pelos softwares selecionados para aplicar o PLN e a rede SOM. As tarefas executadas no pré-processamento da base de Dorks são apresentadas na Figura 28.

Figura 28 – Fluxo das tarefas executadas no pré-processamento da base de Dorks



Fonte: o autor (2020).

A seguir, descrevem-se as tarefas executadas no pré-processamento da base de Dorks.

No software Excel, o hiperlink das Dorks e os atributos “Autor” e “Data” da base foram removidos. Tais atributos não foram considerados na abordagem por não influenciarem a extração de conhecimento da base, nem a execução automática das Dorks. Desta forma, a base ficou com 2 atributos restantes: “Dork” e “Categoria”.

Em seguida, criou-se uma legenda numérica para cada categoria presente na base de Dorks, e substituiu-se o valor textual pela legenda numérica, possibilitando que a informação fosse utilizada pela SOM. Na Tabela 23, apresentam-se as categorias da base de Dorks e suas respectivas legendas.

Tabela 23 – Categoria de Dorks e suas respectivas legendas

Categorias da base de Dorks	**Legenda**
Footholds	1
Files Containing Usernames	2
Sensitive Directories	3
Web Server Detection	4
Vulnerable Files	5
Vulnerable Servers	6
Error Messages	7
Files Containing Juicy Info	8
Files Containing Passwords	9
Sensitive Online Shopping Info	10
Network of Vulnerability Data	11
Pages Containing Login Portals	12
Various Online Devices	13
Advisories and Vulnerabilities	14

Fonte: o autor (2020).

Após definir a legenda, no atributo “Categoria” da base, os valores textuais foram substituídos por numéricos. Feito isso, o próximo passo foi tornar as Dorks específicas em genéricas, ou seja, modificar as Dorks específicas para um determinado site, para que todas possam ser executadas em qualquer outra página da internet.

Para isso, pesquisou-se na base quais Dorks possuíam em sua composição o parâmetro “Site”. Segundo Pan *et al*. (2012), o parâmetro “Site” nas Dorks é um operador avançado do Google utilizado para redirecionar a busca da Dork para um domínio específico, como **site: “.gov.br”** por exemplo.

Após encontrar as Dorks que continham o parâmetro “Site”, todas foram modificadas, removendo esse parâmetro. Dentre elas, encontraram-se Dorks específicas para: sites de Proxy, Google Drive, GitHub, MediaFire, Dropbox e SourceForge. Em seguida, testaram-se formas de dividir a Dork para acrescentar novos atributos na base definida na fase C, tornando-a apta para ser utilizada nos experimentos com a rede SOM. Foram executados 4 pré-processamentos utilizando a linguagem Python, e todos foram avaliados com a aplicação da SOM.

No **primeiro pré-processamento**, converteu-se a Dork inteira em um valor numérico, e esse valor foi acrescentado como um novo atributo, deixando a base com um total de 3. A lógica desse algoritmo é apresentada no Apêndice D. Como a aplicação da SOM não trouxe o resultado esperado, foi necessário pré-processar a base novamente.

No **segundo pré-processamento**, dividiu-se a Dork por parâmetros utilizando a classificação das palavras que compõe a Dork proposta por Pan *et al*. (2012). Com isso, descobriu-se qual Dork possuía maior quantidade de parâmetros em sua composição. Assim, foi possível criar na base de Dorks essa mesma quantidade de atributos para receber as Dorks divididas em parâmetros. A lógica desse algoritmo é apresentada no Apêndice E. Como a aplicação da rede SOM não trouxe o resultado esperado, foi necessário pré-processar a base novamente.

No **terceiro pré-processamento**, dividiu-se a Dork por palavras aplicando a tokenização por PLN. Com isso, descobriu-se qual Dork possuía maior quantidade de palavras em sua composição. Assim, foi possível criar na base de Dorks essa mesma quantidade de atributos para receber as Dorks divididas em palavras. A lógica desse algoritmo é apresentada no Apêndice F. Como a aplicação da rede SOM não trouxe o resultado esperado, foi necessário pré-processar a base novamente.

No **quarto pré-processamento**, dividiu-se a Dork por caracteres aplicando a tokenização por PLN, e, em seguida, todos os caracteres foram selecionados. Com isso, descobriu-se qual Dork possuía maior quantidade de caracteres em sua composição. Assim, foi possível criar na base de Dorks essa mesma quantidade de atributos para receber as Dorks divididas em palavras. A lógica desse algoritmo é apresentada no Apêndice G.

Após a execução do quarto pré-processamento, descobriu-se que a maior quantidade de caracteres presentes em uma Dork na base era de 148. Assim, acrescentou-se os 148 atributos na base de Dorks, chamados de Carac01, Carac02, e assim sucessivamente até Carac148.

Percebeu-se, neste momento, que poucas Dorks possuíam mais de 100 caracteres em sua composição. Em consequência disso, os últimos atributos da base tinham, em sua maioria, valores como 0 (Zero). Para diminuir essa quantidade de valores, foram identificadas quais Dorks possuíam mais de 100 caracteres, utilizando outro algoritmo em Python.

As Dorks identificadas no algoritmo foram tratadas como **outliers**. Essas Dorks possuíam mais de 100 caracteres por dois motivos principais: Dorks compostas e URL's. As Dorks compostas são Dorks que possuem em sua *String* mais de uma Dork. Já as URLs são links para vulnerabilidades específicas em determinados sites. As Dorks que se tratavam de URL's foram removidas, pois não haveria forma de torná-las genéricas para, assim, executá-las de forma automática em outras páginas web. Já as Dorks compostas foram divididas em Dorks menores, e, em seguida, acrescidas na base em suas respectivas categorias.

No total, encontraram-se 102 outliers, sendo que 93 foram tratados, e 9, removidos. Os *outliers* são descritos no Apêndice C deste trabalho. Desta forma, a base passou a possuir uma nova quantidade de Dorks (com mais 217), totalizando 4.428 Dorks na base. Na Tabela 24, apresenta-se a quantidade inicial de Dorks na base e a final, após o tratamento dos outliers.

Tabela 24 – Base de Dorks após o tratamento dos outliers

Categorias da base de Dorks	Quantidade de Dorks Inicial	Quantidade de Dorks Final
Footholds	89	104
Files Containing Usernames	20	20
Sensitive Directories	268	271
Web Server Detection	125	129
Vulnerable Files	58	63
Vulnerable Servers	94	112
Error Messages	105	113
Files Containing Juicy Info	433	450
Files Containing Passwords	212	243
Sensitive Online Shopping Info	9	9
Network of Vulnerability Data	72	80
Pages Containing Login Portals	398	445
Various Online Devices	349	393
Advisories and Vulnerabilities	1979	1996
Total	4211	4428

Fonte: o autor (2020).

Após o tratamento dos outliers, removeram-se as *stopwords* das Dorks com a aplicação de PLN, a fim de diminuir o tamanho das Dorks e, consequentemente, a quantidade de atributos com uma grande quantidade de valores nulos.

Para a remoção, definiram-se 40 caracteres especiais como *stopwords* a serem removidas. As *stopwords* foram definidas com a NLTK, uma vez que essa biblioteca disponibiliza uma lista de caracteres especiais que podem ser removidos de um determinado texto. As *stopwords* removidas foram as seguintes:

,':;"’!?”()`@~/|*[]^_.+\#%¨¬&©ºª}{£¢§-⌂

A síntese dos pré-processamentos realizados é apresentada na Tabela 25.

Tabela 25 – Síntese dos pré-processamentos realizados

ID	Pré-processamento	Função	Quantidade de novos atributos	Aplicação da SOM funcionou?
1º	Conversão da Dork inteira para ASCII.	Ord()	01	Não
2º	Divisão da Dork por parâmetros e conversão para ASCII.	Split (“ “)	09	Não
3º	Tokenização da Dork por PLN e conversão para ASCII.	Nltk.word_ tokenize	70	Não
4º	Tokenização da Dork por PLN, divisão da Dork por caractere, remoção de *outliers* e *stopwords*, e conversão para ASCII.	Nltk.word_ tokenize	**84**	**Sim**

Fonte: o autor (2020).

Desta forma, aplicando o quarto pré-processamento, a base passou a possuir 84 novos atributos. Acrescentando o atributo “Dork”, que contém a Dork textual, e o atributo “Categoria”, que contém a legenda numéricao total de atributos passou a ser 86.

FASE F – TRANSFORMAÇÃO DA BASE DE DORKS

Após pré-processar a base, foi necessário converter os caracteres das Dorks para um valor numérico, possibilitando assim a aplicação da SOM para extrair conhecimento da base, e tornando-a possível de ser executada automaticamente.

Para isso, teve-se como suporte o estudo de Guo *et al*. (2018), que converte caracteres para seu valor numérico em ASCII para detectar vulnerabilidades do tipo “Estouro de Memória”. Para realizar essa conversão, utilizou-se a função “Ord( )” da linguagem Python, mesma função empregada no estudo de Guo *et al*. (2018).

A Dork: **inurl:/phpmyadmin/index.php?db=** foi dividida em 25 caracteres convertidos para valor numérico em ASCII, ficando da seguinte forma:

105	110	117	114	108	112	104	112	109	121	97	100	109
	105	110	105	110	100	101	120	112	104	112	100	98

Lembrando-se que todos os caracteres especiais, definidos como *stopwords*, foram removidos com a aplicação da PLN.

FASE G – APLICAÇÃO DA SOM NA BASE DE DORKS

Após pré-processar e transformar a base de Dorks, foi aplicada a SOM:

a) pela capacidade de execução de Dorks com a ferramenta “DorkMe”, selecionada na fase D **(Seleção e Avaliação das Ferramentas OSINT)**. A ferramenta “DorkMe” mostrou possuir um limite, pois só conseguiu executar o arquivo .txt com 1.250 Dorks, não tornando possível, portanto, executar toda a base de Dorks de forma automática.

Também não foi possível executar a base dividida por categorias, pois a categoria “Advisories and Vulnerabilities” possui um total de 1.979 Dorks, quantidade esta que é superior ao limite da ferramenta “DorkMe”.

b) pela necessidade de extração de conhecimento da base de Dorks. A aplicação da SOM não apenas segmentou as Dorks em agrupamentos com quantidade menores de Dorks, como também proporcionou a extração de conhecimento da base com a formação de agrupamentos por similaridade. Tal resultado propiciou uma nova classificação para as Dorks, apoiando assim a execução da abordagem aqui proposta de forma automática.

Apesar de a SOM conseguir executar a base com as Dorks divididas por caracteres, os agrupamentos gerados na rede possuem uma quantidade de Dorks superior à capacidade da ferramenta selecionada. Desta forma, decidiu-se dividir a base de Dorks por categorias e aplicar a rede SOM em cada uma delas separadamente.

Desta forma, a base de Dorks foi dividida em 14 bases menores, cada uma contendo uma categoria. Além disso, removeu-se o atributo “Categoria” de cada uma das 14 bases, de modo que todas passaram a possuir um total de 85 atributos: o atributo textual Dork contendo a Dork propriamente dita e os 84 atributos numéricos (Carac01 até Carac84).

Para executar a SOM nas 14 bases de Dorks, definiu-se a dimensão de mapa com 225 neurônios, ou seja, um mapa 15x15, e vizinhança topológica hexagonal. Além disso, os parâmetros usados na fase de treinamento foram: número de épocas (iterações) igual a 3000 e taxa de aprendizado igual a 0,5 (KASKI; KOHONEN, 1997).

FASE H – ANÁLISE DOS RESULTADOS

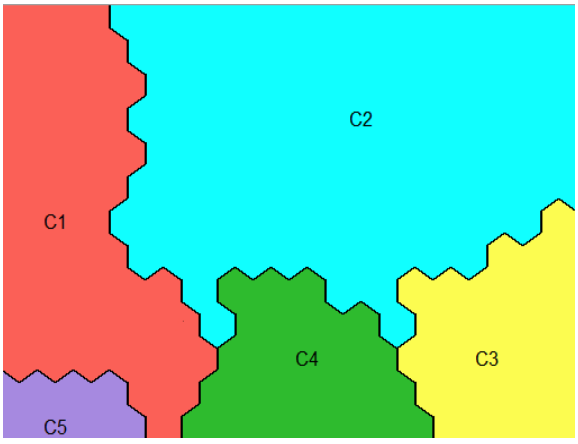
Nesta fase, analisaram-se os resultados gerados da aplicação da SOM nas 14 categorias da base de Dorks, que foi selecionada na fase C, pré-processada na fase E e transformada na fase F. Descreve-se, a seguir, a análise dos resultados da aplicação da SOM nas 14 categorias/bases de Dorks.

BASE 01 – FOOTHOLDS

Esta categoria possui um total de 104 Dorks, que buscam páginas web com algum rastro de vulnerabilidades em sua estrutura. Das 104 Dorks, identificou-se que a maior possui um total de 65 caracteres. Isso significa que, nesta base, os atributos a partir do **Carac66** possuem valores 0 (zeros). Desta forma, removeram-se os atributos **Carac66** até **Carac84**, totalizando uma remoção de 19 atributos.

O Mapa gerado pela SOM e aplicado nesta base é apresentado na Figura 29.

Figura 29 – Mapa gerado pela aplicação da rede SOM na base 01



Fonte: o autor (2020).

A aplicação da rede SOM gerou cinco agrupamentos na base “Footholds”. Na tabela 26, apresentam-se as características de cada um deles.

Tabela 26 – Características do mapa gerado pela aplicação da rede SOM na base 01

Agrupamento	Cor	Dorks	Registros (%)	Vulnerabilidade
C1	Vermelho	26	25%	Sistemas com shell e senhas expostas por meio da palavra “passwords”
C2	Azul	49	47,12%	Páginas web Php com ferramentas dos tipos: editores e testes
C3	Amarelo	15	14,42%	Dorks menores compostas por até 18 caracteres
C4	Verde	9	8,65%	Serviços de e-mail em Php
C5	Lilás	5	4,81%	Tecnologias web com componentes Nett2 e Ajax

Fonte: o autor (2020).

Observa-se, na Tabela 26, que as Dorks abordam vulnerabilidades em serviços utilizados em páginas web, como editores Php, sistemas com shell para codificações de códigos e serviços de e-mail, além de tecnologias como Nett2, Ajax e Php.

BASE 02 – FILES CONTAINING USERNAMES

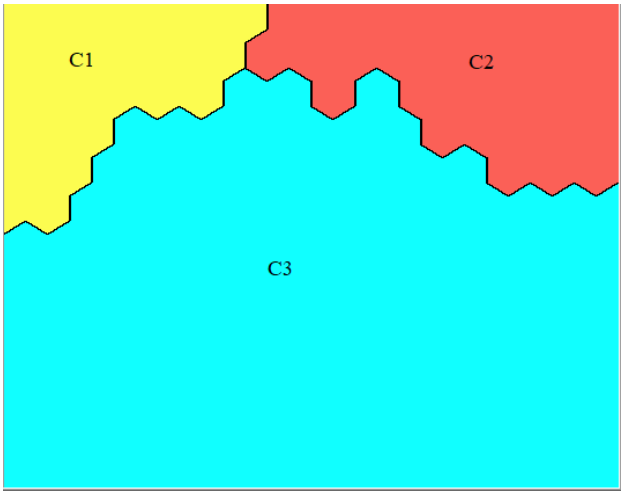
A aplicação da rede SOM nesta base não trouxe resultados satisfatórios devido à baixa quantidade de Dorks presente, um total de 20.

BASE 03 – SENSITIVE DIRECTORIES

Esta categoria possui um total de 271 Dorks, que buscam páginas web com diretórios sensíveis desprotegidos. Das 271 Dorks, identificou-se que a maior possui um total de 67 caracteres. Isso significa que, nesta base, os atributos a partir do **Carac68** possuem valores 0 (zeros). Desta forma, removeram-se os atributos **Carac68** até **Carac84**, totalizando uma remoção de 17 atributos.

O mapa gerado pela SOM e aplicado nesta base é apresentado na Figura 30.

Figura 30 – Mapa gerado pela aplicação da rede SOM na base 03



Fonte: o autor (2020).

A aplicação da rede SOM gerou três agrupamentos na base “Sensitive Directories”. Na tabela 27, apresentam-se as características de cada um deles.

Tabela 27 – Características do mapa gerado pela aplicação da rede SOM na base 03

Agrupamento	**Cor**	**Dorks**	**Registros (%)**	**Vulnerabilidade**
C1	Amarelo	35	12,91%	Plugins, conectores e diretórios de upload
C2	Vermelho	45	16,61%	Diretórios não indexados
C3	Azul	191	70,48%	Diretórios de backup, configurações administrativas e de inclusões Wordpress

Fonte: o autor (2020).

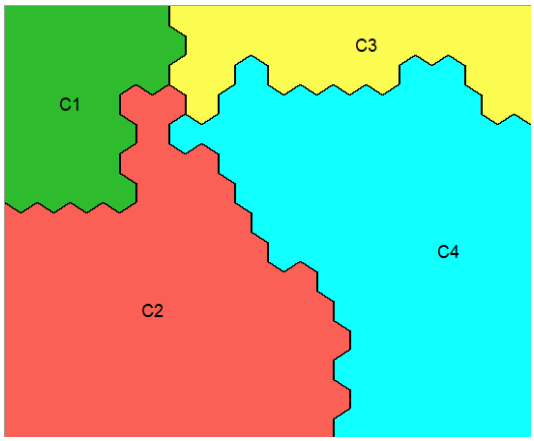
Observa-se, na Tabela 27, que as Dorks abordam vulnerabilidades em diretórios não indexados presentes em páginas web. Os diretórios buscados pelas Dorks são específicos em uma estrutura de site, como: diretórios de plugins, conectores, uploads, backup, inclusões Wordpress, além de diretórios “admin” para configurações administrativas.

BASE 04 – WEB SERVER DETECTION

Esta categoria possui um total de 129 Dorks, que buscam páginas web com informações desprotegidas sobre seu servidor. Das 129 Dorks, identificou-se que a maior possui um total de 73 caracteres. Isso significa que, nesta base, os atributos a partir do **Carac74** possuem valores 0 (zeros). Desta forma, removeram-se os atributos **Carac74** até **Carac84**, totalizando uma remoção de 11 atributos.

O mapa gerado pela SOM e aplicado nesta base é apresentado na Figura 31.

Figura 31 – Mapa gerado pela aplicação da rede SOM na base 04



Fonte: o autor (2020).

A aplicação da rede SOM gerou quatro agrupamentos na base “Web Server Detection”. Na Tabela 28, apresentam-se as características de cada um deles.

Tabela 28 – Características do mapa gerado pela aplicação da rede SOM na base 04

Agrupamento	**Cor**	**Dorks**	**Registros (%)**	**Vulnerabilidade**
C1	Verde	16	12,40%	Páginas de teste do servidor Apache Tomcat
C2	Vermelho	59	45,74%	Servidores abertos em dispositivos Linux e servidores Windows Server
C3	Amarelo	15	11,63%	Arquivos Wsdl e domcfg.ntf's
C4	Azul	39	30,23%	Microsoft IIS

Fonte: o autor (2020).

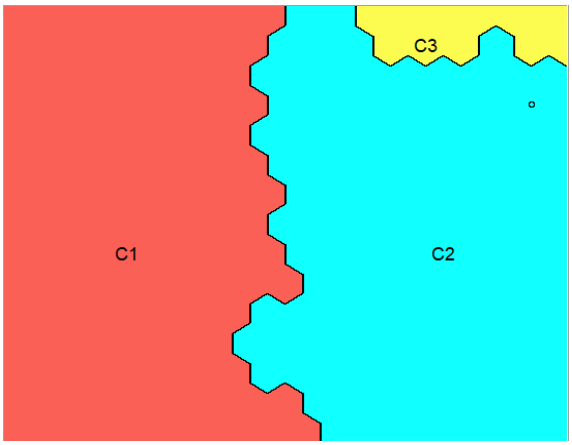
Observa-se, na Tabela 28, que as Dorks indicam vulnerabilidades que permitem identificar o servidor utilizado em uma página web. As vulnerabilidades apresentadas nessas Dorks abordam os servidores Windows, Linux, Apache Tomcat e Microsoft IIS, além de arquivos Wsdl e domcfg.nfts expostos.

BASE 05 – VULNERABLE FILES

Esta categoria possui um total de 63 Dorks, que buscam páginas web com arquivos desprotegidos. Das 63 Dorks, identificou-se que a maior possui um total de 62 caracteres. Isso significa que, nesta base, os atributos a partir do **Carac63** possuem valores 0 (zeros). Desta forma, removeram-se os atributos **Carac63** até **Carac84**, totalizando uma remoção de 22 atributos.

O mapa gerado pela rede SOM e aplicado nesta base é apresentado na Figura 32.

Figura 32 – Mapa gerado pela aplicação da rede SOM na base 05



Fonte: o autor (2020).

A aplicação da rede SOM gerou três agrupamentos na base “Vulnerable Files”. Na Tabela 29, apresentam-se as características de cada um deles.

Tabela 29 – Características do mapa gerado pela aplicação da rede SOM na base 05

Agrupamento	**Cor**	**Dorks**	**Registros (%)**	**Vulnerabilidade**
C1	Vermelho	32	50,8%	Ferramentas Php Setup e Perl Setup
C2	Azul	29	46,03%	Dispositivos “Powered By”
C3	Amarelo	2	3,17%	Bootstrap

Fonte: o autor (2020).

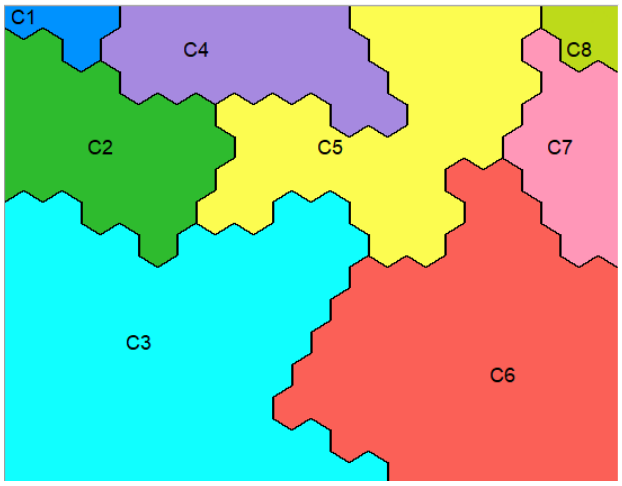
Observa-se, na Tabela 29, que as Dorks abordam vulnerabilidades que permitem identificar arquivos desprotegidos em uma página web. As vulnerabilidades apresentadas nessas Dorks indicam arquivos como: phpSetup, PerlSetup, arquivos de configuração de dipositivos personalizados e arquivos Bootstrap expostos.

BASE 06 – VULNERABLE SERVERS

Esta categoria possui um total de 112 Dorks, que buscam páginas web com arquivos desprotegidos. Das 112 Dorks, identificou-se que a maior possui um total de 80 caracteres. Isso significa que, nesta base, os atributos a partir do **Carac81** possuem valores 0 (zeros). Desta forma, removeram-se os atributos **Carac81**, **Carac82**, **Carac83** e **Carac84**, totalizando uma remoção de 4 atributos.

O mapa gerado pela rede SOM e aplicado nesta base é apresentado na Figura 33.

Figura 33 – Mapa gerado pela aplicação da rede SOM na base 06



Fonte: o autor (2020).

A aplicação da rede SOM gerou oito agrupamentos na base “Vulnerable Servers”. Na Tabela 30, apresentam-se as características de cada um deles.

Tabela 30 – Características do mapa gerado pela aplicação da rede SOM na base 06

Agrupamento	**Cor**	**Dorks**	**Registros (%)**	**Vulnerabilidade**
C1	Azul-escuro	2	1,79%	Mensagens de *phishing*
C2	Verde-escuro	9	8,04%	Servidores de e-commerce e servidores com SMTP
C3	Azul-claro	39	34,82%	PhpMyAdmin e JooManager
C4	Lilás	11	9,82%	Scanner (Listener) de diretórios Php
C5	Amarelo	13	11,61%	Open SSL
C6	Vermelho	29	25,89%	Arquivos .cgi e *Strings* (“ “) específicas em URLs
C7	Rosa	7	6,25%	Arquivos RpSys
C8	Mostarda	2	1,79%	Arquivos R57.Shell

Fonte: o autor (2020).

Observa-se, na Tabela 30, que as Dorks abordam vulnerabilidades que permitem identificar servidores utilizados em páginas web. Tais vulnerabilidades apresentam arquivos e diretórios que possibilitam a edição de suas informações sem a necessidade de alguma autenticação.

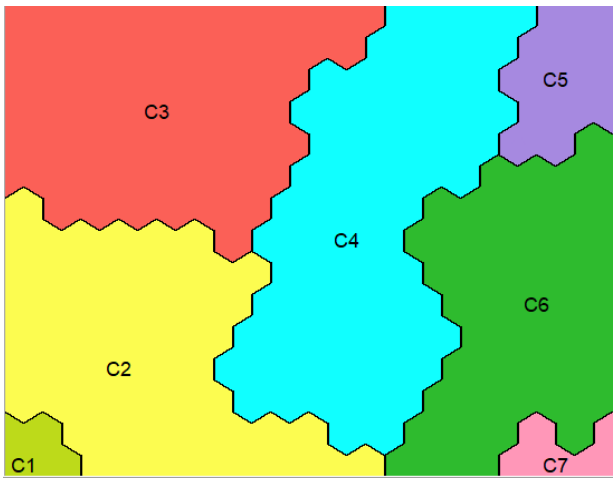
BASE 07 – ERROR MESSAGES

Esta categoria possui um total de 63 Dorks, que buscam páginas web com arquivos desprotegidos. Das 63 Dorks, identificou-se que a maior possui um total de 79 caracteres. Isso

significa que, nesta base, os atributos a partir do **Carac80** possuem valores 0 (zeros). Desta forma, removeram-se os atributos **Carac80**, **Carac81**, **Carac82**, **Carac83** e **Carac84**, totalizando uma remoção de 5 atributos.

O mapa gerado pela rede SOM e aplicado nesta base é apresentado na Figura 34.

Figura 34 – Mapa gerado pela aplicação da rede SOM na base 07



Fonte: o autor (2020).

A aplicação da rede SOM gerou sete agrupamentos na base “Error Messages”. Na Tabela 31, apresentam-se as características de cada um deles.

Tabela 31 – Características do mapa gerado pela aplicação da rede SOM na base 07

Agrupamento	**Cor**	**Dorks**	**Registros (%)**	**Vulnerabilidade**
C1	Mostarda	3	2,65%	SQL Server Driver Exception
C2	Amarelo	21	18,58%	Mensagens de warning em requisições a servidores web
C3	Vermelho	38	33,63%	Mensagens de acesso negado
C4	Azul	24	21,24%	Mensagens de erro em Querys e Sintaxes
C5	Lilás	7	6,19%	Mensagens de erro Oracle “ORA”
C6	Verde	17	15,04%	Mensagens de erro de Sitebuilder
C7	Rosa	3	2,65%	Mensagens de erro com arquivos de extensão .cfm

Fonte: o autor (2020).

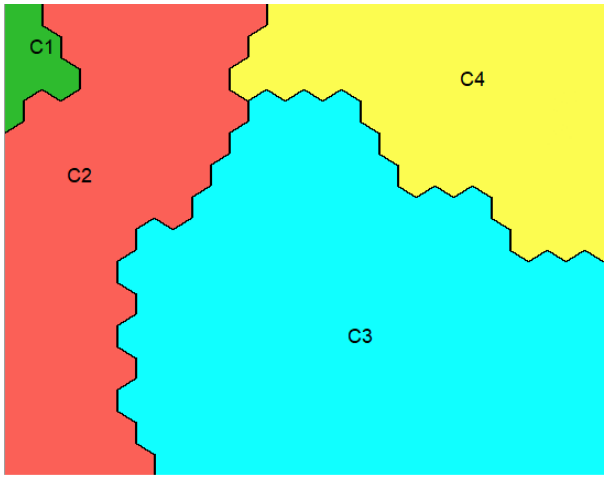
Observa-se, na Tabela 31, que as Dorks abordam vulnerabilidades que permitem explorar mensagens de erro em páginas web; mensagens estas que, de alguma forma, acabam exibindo informações e arquivos sensíveis.

BASE 08 – FILES CONTAINING JUICY INFO

Esta categoria possui um total de 450 Dorks, que buscam páginas web com arquivos desprotegidos. Das 450 Dorks, identificou-se que a maior possui um total de 75 caracteres. Isso significa que, nesta base, os atributos a partir do **Carac76** possuem valores 0 (zeros). Desta forma, removeram-se os atributos **Carac76** até **Carac84**, totalizando uma remoção de 9 atributos.

O mapa gerado pela rede SOM e aplicado nesta base é apresentado na Figura 35.

Figura 35 – Mapa gerado pela aplicação da rede SOM na base 08



Fonte: o autor (2020).

A aplicação da rede SOM gerou quatro agrupamentos na base “Files Containing Juicy Info”. Na Tabela 32, apresentam-se as características de cada um deles.

Tabela 32 – Características do mapa gerado pela aplicação da rede SOM na base 08

Agrupamento	Cor	Dorks	Registros (%)	Vulnerabilidade
C1	Verde	6	1,33%	Querys de requisições SQL Dump
C2	Vermelho	135	30,00%	Arquivos sobre status de servidores e arquivos com endereços de e-mail.
C3	Azul	228	50,67%	Arquivos com extensão .log, .txt, .dat e .asp, com exclusão do GitHub -git
C4	Amarelo	81	18,00%	Arquivos com extensão proj e arquivos Netscape

Fonte: o autor (2020).

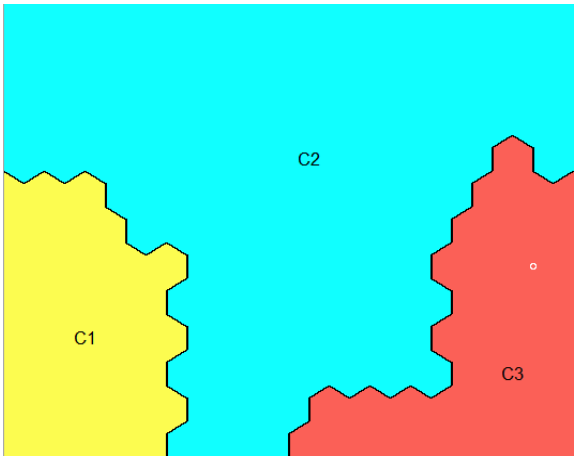
Observa-se, na Tabela 32, que as Dorks abordam vulnerabilidades que permitem explorar arquivos desprotegidos com informações sobre outros sistemas em páginas web. Essas vulnerabilidades são de diversas extensões, abrangendo tecnologias como SQL e Netscape.

BASE 09 – FILES CONTAINING PASSWORDS

Esta categoria possui um total de 243 Dorks, que buscam páginas web com arquivos desprotegidos. Das 243 Dorks, identificou-se que a maior possui um total de 64 caracteres. Isso significa que, nesta base, os atributos a partir do **Carac65** possuem valores 0 (zeros). Desta forma, removeram-se os atributos **Carac66** até **Carac84**, totalizando uma remoção de 20 atributos.

O mapa gerado pela rede SOM e aplicado nesta base é apresentado na Figura 36.

Figura 36 – Mapa gerado pela aplicação da rede SOM na base 09



Fonte: o autor (2020).

A aplicação da rede SOM gerou três agrupamentos na base “Files Containing Passwords”. Na Tabela 33, apresentam-se as características de cada um deles.

Tabela 33 – Características do mapa gerado pela aplicação da rede SOM na base 09

Agrupamento	Cor	Dorks	Registros (%)	Vulnerabilidade
C1	Amarelo	46	18,93%	Arquivos de senhas com extensão XLS e XLSX
C2	Azul	151	62,14%	Arquivos não indexados de diversas extensões com: password, pwd e passwd
C3	Vermelho	46	18,93%	Arquivos com as palavras login e logon

Fonte: o autor (2020).

Observa-se, na Tabela 33, que as Dorks abordam vulnerabilidades que permitem explorar arquivos desprotegidos com informações sobre senhas. Tais vulnerabilidades buscam arquivos de variados tipos, como Xls (Arquivos Excel) e arquivos Passwd (Arquivos de senha), além de procurarem por palavras como “login” e “logon”.

BASE 10 – SENSITIVE ONLINE SHOPPING INFO

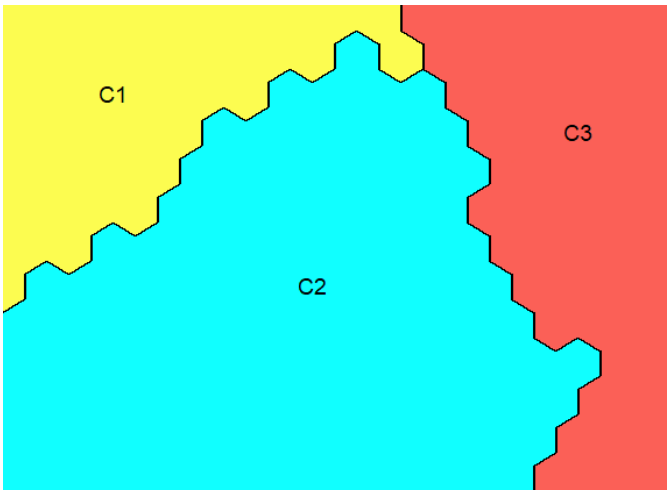
A aplicação da rede SOM nesta base não trouxe resultados satisfatórios devido à baixa quantidade de Dorks presente, um total de 9.

BASE 11 – NETWORK OF VULNERABILITY DATA

Esta categoria possui um total de 80 Dorks, que buscam páginas web com arquivos desprotegidos. Das 80 Dorks, identificou-se que a maior possui um total de 81 caracteres. Isso significa que, nesta base, os atributos a partir do **Carac82** possuem valores 0 (zeros). Desta forma, removeram-se os atributos **Carac82**, **Carac83** e **Carac84**, totalizando uma remoção de 3 atributos.

O mapa gerado pela rede SOM e aplicado nesta base é apresentado na Figura 37.

Figura 37 – Mapa gerado pela aplicação da rede SOM na base 11



Fonte: o autor (2020).

A aplicação da rede SOM gerou três agrupamentos na base “Network of Vulnerability Data”. Na Tabela 34, apresentam-se as características de cada deles.

Tabela 34 – Características do mapa gerado pela aplicação da rede SOM na base 11

Agrupamento	Cor	Dorks	Registros (%)	Vulnerabilidade
C1	Amarelo	16	20,00%	Mensagens de erro e mensagens de IPSwitchs
C2	Azul	41	51,25%	Mensagens com catálogos, client info e admin info
C3	Vermelho	23	28,75%	Diretórios Munin/html

Fonte: o autor (2020).

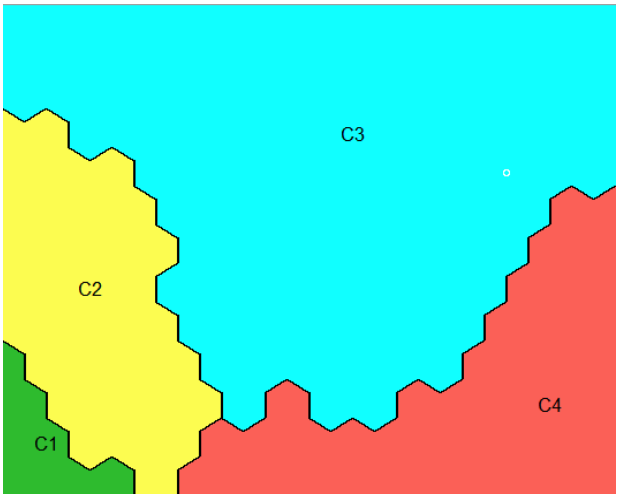
Observa-se, na Tabela 34, que as Dorks abordam vulnerabilidades que permitem explorar mensagens e diretórios de serviços de rede, os quais estão interligados com uma determinada página web. Tais vulnerabilidades buscam arquivos com mensagens de erros em serviços de rede, além de diretórios Munin em tecnologias HTML.

BASE 12 – PAGES CONTAINING LOGIN PORTALS

Esta categoria possui um total de 445 Dorks, que buscam páginas web com arquivos desprotegidos. Das 445 Dorks, identificou-se que a maior possui um total de 79 caracteres. Isso significa que, nesta base, os atributos a partir do **Carac80** possuem valores 0 (zeros). Desta forma, removeram-se os atributos **Carac80**, **Carac81**, **Carac82**, **Carac83** e **Carac84**, totalizando uma remoção de 5 atributos.

O mapa gerado pela rede SOM e aplicado nesta base é apresentado na Figura 38.

Figura 38 – Mapa gerado pela aplicação da rede SOM na base 12



Fonte: o autor (2020).

A aplicação da rede SOM gerou quatro agrupamentos na base “Pages Containing Login Portals”. Na Tabela 35, apresentam-se as características de cada um deles.

Tabela 35 – Características do mapa gerado pela aplicação da rede SOM na base 12

Agrupamento	Cor	Dorks	Registros (%)	Vulnerabilidade
C1	Verde	18	4,04%	Mensagens de erro via Microsft IE 5.5 e parâmetros –edu e -help
C2	Amarelo	77	17,30%	Controladores Mikrotik e Honeywell
C3	Azul	268	60,22%	Diretórios admin, login, logon e client log
C4	Vermelho	82	18,43%	Arquivos de login com extensões: .asp, .aspx, .cgi e .php

Fonte: o autor (2020).

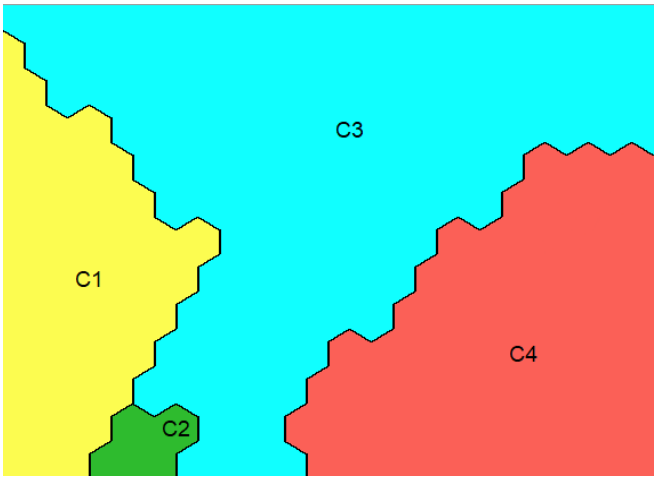
Observa-se, na Tabela 35, que as Dorks abordam vulnerabilidades que permitem explorar páginas web contendo homepages (com login – usuário e senha) desatualizadas ou com erros e com uma estrutura para o uso de sessões. Essas informações sensíveis são exibidas por meio de mensagens de erro, diretórios desprotegidos e arquivos de configurações de dispositivos de rede.

BASE 13 – VARIOUS ONLINE DEVICES

Esta categoria possui um total de 393 Dorks, que buscam páginas web com arquivos desprotegidos. Das 393 Dorks, identificou-se que a maior possui um total de 78 caracteres. Isso significa que, nesta base, os atributos a partir do **Carac79** possuem valores 0 (zeros). Desta forma, removeram-se os atributos **Carac79**, **Carac80**, **Carac81**, **Carac82**, **Carac83** e **Carac84**, totalizando uma remoção de 6 atributos.

O mapa gerado pela rede SOM e aplicado nesta base é apresentado na Figura 39.

Figura 39 – Mapa gerado pela aplicação da rede SOM na base 13



Fonte: o autor (2020).

A aplicação da rede SOM gerou quatro agrupamentos na base “Various Online Devices”. Na Tabela 36, apresentam-se as características de cada um deles.

Tabela 36 – Características do mapa gerado pela aplicação da rede SOM na base 13

Agrupamento	**Cor**	**Dorks**	**Registros (%)**	**Vulnerabilidade**
C1	Amarelo	74	18,83%	Servidores web e gateways
C2	Verde	4	1,02%	Database Schema
C3	Azul	201	51,15%	Impressoras, servidores de e-mail, manuais e guias
C4	Vermelho	114	29,01%	NetFone e câmeras

Fonte: o autor (2020).

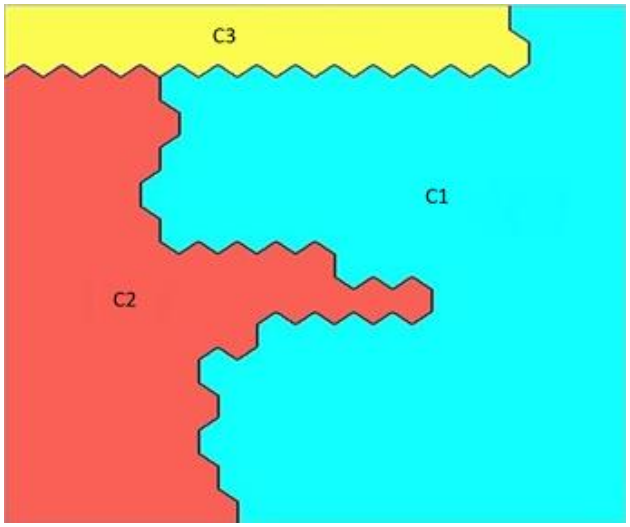
Observa-se, na Tabela 36, que as Dorks abordam vulnerabilidades que permitem explorar dispositivos on-line desprotegidos e expostos em páginas web. Tais vulnerabilidades buscam dispositivos como: servidores web, gateways, base de dados, servidores de e-mail, impressoras e câmeras.

BASE 14 – ADVISORIES AND VULNERABILITIES

Esta categoria possui um total de 1.996 Dorks, que buscam páginas web com arquivos desprotegidos. Das 1.996 Dorks, identificou-se que a maior possui um total de 84 caracteres. Desta forma, não foi necessário remover nenhum atributo desta base.

O mapa gerado pela rede SOM e aplicado nesta base é apresentado na Figura 40.

Figura 40 – Mapa gerado pela aplicação da rede SOM na base 14



Fonte: o autor (2020).

A aplicação da rede SOM gerou três agrupamentos na base “Advisories and Vulnerabilities”. Na Tabela 37, apresentam-se as características de cada um deles.

Tabela 37 – Características do mapa gerado pela aplicação da rede SOM na base 14

Agrupamento	**Cor**	**Dorks**	**Registros (%)**	**Vulnerabilidade**
C1	Azul	1092	54,71%	Dispositivos on-line
C2	Vermelho	622	31,16%	Requisições URL
C3	Amarelo	282	14,13%	Requisições URL compostas

Fonte: o autor (2020).

Observa-se, na Tabela 37, que as Dorks abordam vulnerabilidades que permitem anúncios e outras mensagem em páginas web. Tais vulnerabilidades buscam dispositivos on-line e requisições URL para, assim, procurar informações sensíveis.

Após a geração dos 12 mapas, avaliou-se sua qualidade por meio dos erros de quantização e topográfico. Na Tabela 38, apresenta-se o resultado das métricas dos 12 mapas gerados na fase G.

Tabela 38 – Resultados das métricas dos 12 mapas gerados pela rede SOM

Base	EQ Valor Min	EQ Valor Max	EQ Quantidade de Nós Acima de 1	EQ Percentual de Nós Acima de 1	Erro Topográfico
01	0,0218206	6,78658	25	11,1%	0.03846154
03	0,0375418	10,1289	39	17,3%	0.04059041
04	0,0950071	10,7852	22	09,7%	0.003690037
05	0,0325482	1,14028	23	10,2%	0.03174603
06	0,0458371	2,41803	41	18,2%	0.008928571
07	0,0322011	10,3207	47	20,8%	0.01769912
08	0,0990822	77,5008	75	33,3%	0.008849558
09	0,0879076	17,6479	44	19,5%	0.008230453
11	0,0835323	4,21733	22	09,7%	0.0125
12	0,126779	34,8612	56	24,8%	0.006741573
13	0,0205212	55,4796	57	25,3%	0.0178117
14	0,0202605	62,339	13	05,7%	0.01703407

Fonte: o autor (2020).

Analisando a Tabela 38, percebe-se que os 12 mapas gerados pela rede SOM apresentaram erro topográfico próximo de 0 (zero). Isso significa que a topologia dos dados de entrada foi preservada, ou seja, que todos os nós representaram bem o vetor de entrada inicial.

O erro de quantização para os doze mapas apresentou valores mínimos próximos a zero, o que mostra a qualidade dos dados dos vetores de entrada. Percebeu-se que os mapas com Dorks em que o erro de quantização ficou acima de 1 (um) tinham caracteres especiais não previstos nas *stopwords* definidas pela biblioteca NLTK. Esses caracteres possuem valores diferentes do restante dos caracteres alfanuméricos, assumindo um valor distante dos demais caracteres convertidos para ASCII.

Pode-se considerar como bons os erros obtidos nos 12 mapas, uma vez que existe um *trade-off* entre as métricas: erro de quantização e erro topográfico; quanto menor for o erro de quantização, maior será o erro topográfico, isso acontece porque, para obter um erro de quantização menor, basta apenas aumentar o número de neurônios no mapa. Além disso, quanto maior for o mapa, maior será a probabilidade de que o neurônio vencedor e o segundo neurônio vencedor não sejam adjacentes, aumentando o erro topográfico.

FASE I – ADIÇÃO DAS NOVAS INFORMAÇÕES NA BASE DE DORKS

Nesta fase, adicionou-se o conhecimento obtido dos resultados da aplicação da rede SOM descritos na fase H na base de Dorks definida na fase C. Para isso, foi criado um novo atributo chamado de *cluster* e inseriu-se um valor para as Dorks, com base nos agrupamentos definidos pela SOM. Na Tabela 39, descreve-se uma amostra dessa adição com a base 14 (Advisories and Vulnerabilities), com uma nova classificação das Dorks. Nela, também consta a quantidade de Dorks que cada agrupamento possui, sua identificação e legenda.

Tabela 39 – Nova classificação da rede SOM na base 14 (Advisories and Vulnerabilities)

Categoria	SOM	Qntd. Dorks	Cluster	Legenda
Advisories and Vulnerabilities	Dispositivos on-line	1092	C1	01
	Requisições URL	622	C2	02
	Requisições URL compostas	282	C3	03

Fonte: o autor (2020).

Por fim, para cada base, exportou-se da ferramenta Viscovery SOMine os agrupamentos de Dorks identificados para arquivos txt. Geraram-se 53 arquivos txt no total, sendo que todos foram utilizados na fase J para validar a abordagem proposta neste trabalho.

4.3 EXPERIMENTOS COMPUTACIONAIS DO TERCEIRO GRUPO DA ABORDAGEM

A seguir, apresentam-se os resultados dos experimentos computacionais do terceiro grupo da abordagem, composto pela fase J.

FASE J – VALIDAÇÃO DA EXECUÇÃO AUTOMÁTICA DE DORKS

Nesta fase, as Dorks foram executadas com as informações descobertas com a aplicação da rede SOM na fase I e com a ferramenta definida na fase C.

Para validação da execução automática de Dorks, realizou-se uma comparação entre as execuções manual e automática, utilizando critérios de avaliação baseados no estudo de Bae, Lim e Cho (2016), que são: Tempo total de execução, Tempo de execução por Dork; Quantidade de vulnerabilidades encontradas e; Média em segundos para cada vulnerabilidade encontrada.

O Tempo total de execução e o tempo de execução por Dork são critérios utilizados para medir o tempo das execuções, automática e manual, e assim, avaliar qual é a mais rápida. Já o critério quantidade de vulnerabilidades encontradas mede o desempenho de cada execução, avaliando quantas vulnerabilidades cada execução consegue encontrar.

Por fim, o critério Média em segundos para cada vulnerabilidade encontrada, representa quantos segundos cada execução demora para encontrar uma vulnerabilidade. Desta forma, é possível avaliar o tempo das duas execuções por cada vulnerabilidade encontrada.

Para calcular a Média em segundos para cada vulnerabilidade encontrada, dividiu-se o valor do Tempo total de execução pela Quantidade de vulnerabilidades encontradas.

Para validar a abordagem, dois agrupamentos gerados pela SOM foram selecionados: o maior e o menor das 14 categorias. O menor agrupamento foi o C1 da base 06 (Vulnerable Servers), com um total de 2 Dorks, que abordam mensagens de *phishing*. Já o maior agrupamento foi o C1 da base 14 (Advisories and Vulnerabilities), com um total de 1.092 Dorks sobre anúncios provenientes de dispositivos on-line.

Na Tabela 40, descreve-se o resultado das execuções automática e manual do menor agrupamento selecionado nesta fase.

Tabela 40 – Primeira execução automática e manual de Dorks

Métricas	**Automático**	**Manual**
Tempo total de execução	98 segundos	298 segundos
Tempo de execução por Dork	49 segundos	149 segundos
Quantidade de vulnerabilidades encontradas	50	50
Média em segundos para cada vulnerabilidade encontrada	1,96	5,96

Fonte: o autor (2020).

Já na Tabela 41, descreve-se o resultado das execuções automática e manual do maior agrupamento selecionado nesta fase.

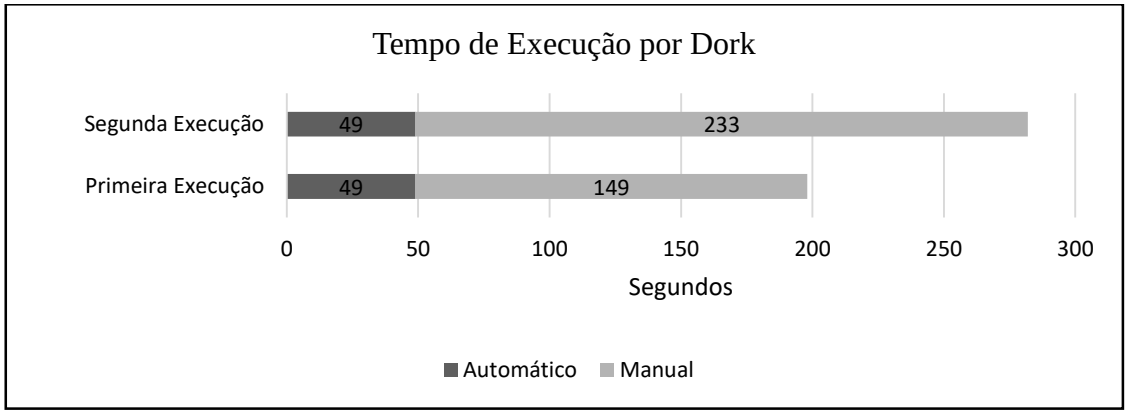
Tabela 41 – Segunda execução automática e manual de Dorks

Métricas	**Automático**	**Manual**
Tempo total de execução	14,8 horas	70,6 horas
Tempo de execução por Dork	49 segundos	233 segundos
Quantidade de vulnerabilidades encontradas	20.810	20.810
Média em segundos para cada vulnerabilidade encontrada	2,5	12,22

Fonte: o autor (2020).

Analisando as Tabelas 41 e 42, percebe-se que a execução automática foi mais rápida que a execução manual. Porém, a tendência é que o tempo de execução cresça conforme o aumento no número de Dorks. As figuras 41 e 42 apresentam o tempo de execução por Dork de ambas as execuções e a média em segundos para cada vulnerabilidade encontrada.

Figura 41 – Tempo de Execução por Dork



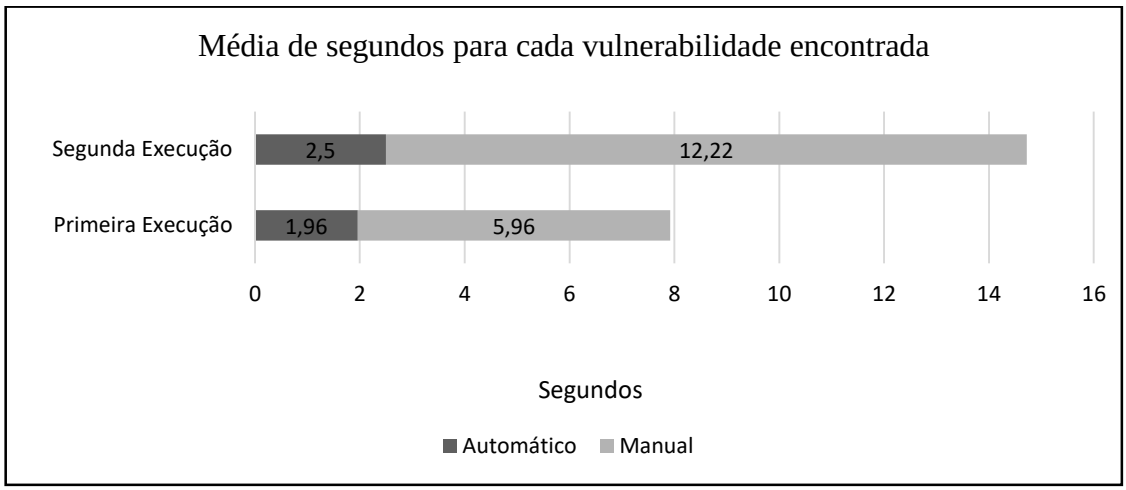
Fonte: o autor (2020).

A figura 41 apresenta o tempo de execução por Dork de ambas as execuções (Automática e Manual). Na primeira execução realizada no menor agrupamento, em que havia apenas 2 Dorks, a execução automática foi 3 vezes mais rápida que a manual. Enquanto a execução automática teve uma duração de 49 segundos, a execução manual teve uma duração de 149 segundos.

Já na segunda execução realizada no maior agrupamento, em que havia 1.092 Dorks, a execução automática foi 4 vezes mais rápida do que a manual. Enquanto a execução automática teve uma duração de 49 segundos, a execução manual teve uma duração de 233 segundos.

A figura 42 apresenta a média em segundos para cada vulnerabilidade encontrada em ambas as execuções.

Figura 42 – Média de segundos para cada vulnerabilidade encontrada



Fonte: o autor (2020).

A figura 42 apresenta a média em segundos para cada vulnerabilidade encontrada em ambas as execuções (Automática e Manual).

A figura 42 apresenta a média em segundos para cada vulnerabilidade encontrada de ambas as execuções (Automática e Manual). Na primeira execução realizada no menor agrupamento, em que havia apenas 2 Dorks, a execução automática foi 3 vezes mais rápida que a manual. Enquanto a execução automática leva 1,96 segundos para encontrar uma vulnerabilidade, a execução manual leva 5,96 segundos.

Já na segunda execução realizada no maior agrupamento, em que havia 1.092 Dorks, a execução automática foi 4 vezes mais rápida do que a manual. Enquanto a execução automática leva 2,5 segundos para encontrar uma vulnerabilidade, a execução manual leva 12,22 segundos.

Essa validação mostrou-se relevante, uma vez que foi possível contrastar as práticas do Google Hacking executadas de formas manual e automática. Além disso, tal validação permitiu identificar a média de vulnerabilidades encontradas por segundo, além do tempo de execução por Dork e o tempo total de execução.

Assim, constatou-se que a utilização de PLN e SOM em uma base de Dorks possibilitou que a ferramenta DorkMe, selecionada na fase D desta abordagem, executasse a prática do Google Hacking de forma automática.

A execução automática do Google Hacking é importante porque, como é realizada de forma mais rápida do que a manual, diminui o risco de as vulnerabilidades serem encontradas

antes por cibercriminosos. Afinal, conforme Chu e Lisitsa (2018), Nagpure e Kurkure (2017) e Hatfield (2018), o tempo é um fator crucial para a detecção de vulnerabilidades.

Em adição, a execução automática do Google Hacking possibilita que o profissional de segurança da informação, que está executando o Pentest, concentre-se em outras tarefas, como a análise dos resultados obtidos. Isso evita que o profissional utilize seu tempo com tarefas repetitivas, como a execução manual do Google Hacking, e permite um melhor aproveitamento de suas habilidades técnicas em outros procedimentos realizados em um Pentest.

5 CONCLUSÃO

Visando a necessidade de se buscar meios para encontrar vulnerabilidades em páginas web, e entendendo o empenho da comunidade acadêmica nesse sentido, desenvolveu-se aqui uma abordagem de Inteligência de Fontes Abertas, com Mapas Auto-Organizáveis de Kohonen e Processamento de Linguagem Natural, para executar Dorks de forma automática, com o objetivo de melhorar a prática do Google Hacking.

A abordagem proposta neste estudo contemplou dez fases, que foram divididas em três grupos. O primeiro foi composto pelas fases A, B, C e D, o segundo grupo, pelas fases E, F, G, H e I, e o terceiro, pela fase J.

A aplicação dessa abordagem permitiu selecionar e avaliar as ferramentas OSINT (15) para executar o Google Hacking de forma automática. Identificadas e selecionadas as ferramentas OSINT, foram criados 5 critérios de avaliação para definir qual ferramenta seria utilizada nas demais fases da abordagem. Dentre as 15 selecionadas, a DorkMe foi a que apresentou melhor desempenho.

Após definir a ferramenta OSINT, selecionou-se a base de Dorks Google Hacking Database para apoiar a execução automática da prática do Google Hacking. Um problema encontrado na base é que as categorias que dividem as Dorks possuem uma quantidade de Dorks maior que a capacidade da ferramenta. Além disso, outro problema descoberto foi a impossibilidade de se aplicar técnicas de IA na base, por causa da pouca quantidade de atributos, isto é, 4.

Desta forma, foi realizada a aplicação de PLN na base para pré-processá-la e adicionar novos atributos. Os resultados obtidos com essa aplicação demonstraram que foi possível pré-processar a base de Dorks e gerar novos atributos para aplicar a rede SOM posteriormente. Com a execução do PLN na base, foram identificados e tratados 102 *outliers* e 40 *stopwords*. Por fim, as Dorks foram divididas por caracteres, e cada um deles se tornou um novo atributo da base, totalizando um total de 84 atributos.

Em seguida, realizou-se a transformação da base de Dorks convertendo os seus caracteres em valores numéricos. Foi executada a conversão do caractere para seu valor em ASCII. Desta forma, a base passou a estar apta para a aplicação de técnicas de IA, como a SOM por exemplo, técnica previamente definida neste trabalho.

Optou-se pela aplicação da rede SOM para segmentar a base em Dorks com quantidades que pudessem ser executadas de forma automática pela ferramenta DorkMe. A aplicação da SOM foi realizada nas 14 categorias de Dorks separadamente.

Apenas nas bases 02 (Files Containing Usernames) e 10 (Sensitive Online Shopping Info), a aplicação da rede SOM não trouxe resultados satisfatórios. Nas demais 12 bases, porém, foi possível gerar agrupamentos de Dorks por similaridade. Todos os 53 agrupamentos gerados pela SOM foram transformados em arquivos .txt, que podem ser executados de forma automática com a ferramenta DorkMe.

Além disso, os mapas gerados pela SOM foram considerados bons, em função dos baixos erros experimentais observados nos resultados. Os erros utilizados como métrica de avaliação dos mapas foram: Erro Topográfico (ET) e Erro de Quantização (EQ).

Em seguida, a abordagem foi validada executando o Google Hacking automático com a ferramenta DorkMe e com dois arquivos .txt extraídos dos agrupamentos gerados pela SOM. Ambos representam o maior e o menor agrupamento das 14 categorias.

O menor agrupamento selecionado foi o C1 da base 06 (Vulnerable Servers), com um total de 2 Dorks, que abordam mensagens de *phishing*. Já o maior agrupamento foi o C1 da base 14 (Advisories and Vulnerabilities), com um total de 1092 Dorks sobre anúncios provenientes de dispositivos on-line.

Os resultados da execução automática foram comparados com suas respectivas execuções manuais. Os critérios de avaliação utilizados reforçam o tempo de execução por se tratar de um fator crítico para detectar vulnerabilidades. Os critérios definidos foram: tempo total de execução, tempo de execução por Dorks, quantidade de vulnerabilidades encontradas e média de vulnerabilidades encontradas por segundo.

Em ambas as execuções, a comparação dos resultados obtidos evidenciou que a execução automática é superior a manual. Além disso, descobriu-se que a diferença entre o tempo das duas execuções só tende a crescer, conforme o aumento de número de Dorks a serem executadas.

Na comparação feita com o menor agrupamento (2 Dorks), a execução automática se mostrou 3 vezes mais rápida. Já na comparação feita com o maior agrupamento (1092 Dorks), a execução automática se mostrou 4 vezes mais rápida.

Desta forma, baseando-se nos resultados apresentados pode-se concluir que a aplicação de PLN para pré-processar a base Google Hacking Database, e a aplicação da rede SOM para extrair conhecimento e gerar agrupamentos de Dorks por similaridade, tornaram a abordagem de Inteligência de Fontes Abertas proposta neste trabalho capaz de executar Dorks de forma automática, melhorando assim o desempenho da prática do Google Hacking. Com isso, considera-se que o objetivo deste estudo foi atingido, e a questão de pesquisa, respondida.

Como principal contribuição deste trabalho para a academia, considera-se o desenvolvimento de uma abordagem de OSINT com PLN e SOM para executar Dorks da base GHDB de forma automática. Em adição, a abordagem desenvolvida possibilita também a realização de agrupamentos de palavras ou sentenças por similaridade em seus caracteres, como foi realizado com as Dorks.

Vale destacar que os fluxogramas representando os testes realizados para desenvolvimento do algoritmo, a fim de aplicar o PLN na base de Dorks, estão descritos nos anexos D, E, F e G deste trabalho. Desta maneira, outros algoritmos poderão ser desenvolvidos baseando-se nesta lógica.

Outra contribuição deste estudo é a revisão sistemática da literatura realizada, uma vez que foram encontradas 248 publicações sobre OSINT, que poderão servir de base para consulta de pesquisadores interessados no assunto aqui tratado. Além disto, a seleção e avaliação de ferramentas OSINT para a execução automática do Google Hacking também podem ser consideradas como uma contribuição deste trabalho.

Em relação às organizações, a contribuição pode residir na utilização da abordagem para identificar vulnerabilidades de forma automática, ganhando mais tempo e, consequentemente, diminuindo o risco de as empresas serem vítimas de crimes cibernéticos. Além disso, tal abordagem poderá apoiar a tomada de decisão de analistas e demais profissionais da área de segurança da informação.

No caso da sociedade, considera-se que este trabalho possibilitará o armazenamento dos dados pessoais de clientes e colaboradores de uma determinada organização em locais mais seguros, pois a abordagem aqui proposta pretende identificar que não há vulnerabilidades já documentadas e relatadas na base do GHDB.

Vale ressaltar que a abordagem proposta para a execução automática do Google Hacking não se limita a páginas web organizacionais, podendo, portanto, ser aplicada em muitas outras, tais como: páginas web de hospitais e outras instituições de saúde, páginas de e-commerce, portais acadêmicos de instituições de ensino e páginas de acesso a armazenamentos em nuvem.

Outro ponto a destacar são as publicações oriundas deste trabalho. Todas as publicações, com exceção da publicação "**Classificação de Fluxos de Dados para Identificação de Anomalias em Honeypots com a Teoria dos Rough Sets",** que apesar de não ser oriunda deste trabalho, aborda a aplicação de Inteligência Artificial na área da Segurança da informação, foram publicadas em revistas e conferências importantes. Estas publicações inserem o Brasil no grupo de países que possuem estudos sobre o tema OSINT, como pode ser observado na revisão sistemática da literatura realizada neste trabalho.

Como continuidade da pesquisa, considera-se a utilização de outras técnicas de IA, aplicadas nas Dorks pré-processadas por PLN, para o desenvolvimento de regras capazes de auxiliar os mecanismos de defesa, como IDS e firewalls, a identificarem a prática do Google Hacking sendo executada ilegalmente em um domínio na web. Além disso, recomenda-se ampliar a revisão sistemática da literatura realizada neste trabalho, para assim, identificar novas lacunas de pesquisa.

Vale destacar as limitações observadas no desenvolvimento deste trabalho como as stopwords definidas na fase E da abordagem. Recomenda-se acrescentar novos caracteres especiais não previstos pela biblioteca NLTK utilizada neste trabalho. Também se ressalta as ferramentas OSINT selecionadas para executar o Google Hacking de forma automática. Sugere-se a busca por outras ferramentas além das descritas neste trabalho, e também a criação de novos critérios de avaliação para descobrir qual ferramenta possui o melhor desempenho.

Os estudos apresentados neste trabalho não tiveram a pretensão de saturar o assunto. Contrário a isso, buscou-se trazer uma contribuição para a academia e para o cenário nacional, propondo uma abordagem de OSINT com PLN e SOM para executar o Google Hacking de forma automática.

5.1 PUBLICAÇÕES DO AUTOR

EVANGELISTA, J. R. G.; SASSI, R. J. ; ROMERO, M.; NAPOLITANO, D. M. R. **Systematic Literature Review to Investigate the Application of Open Source Intelligence (OSINT) with Artificial Intelligence**. Journal of Applied Security Research, p. 1-25, 2020. **QUALIS A4**.

J. R. G. Evangelista; SILVA, E. M. L. ; SASSI, R. J. . **Enriquecimento de Base de Dorks Com Processamento de Linguagem Natural**. Brazilian Journal of Development, v. 6, p. 10763-10780, 2020. **QUALIS B2**.

CORDOVIL, R.S. ; EVANGELISTA, J. R. G. ; SASSI, R. J. ; LIMA, A. S. ; BARBOSA, R. P. D. ; GATTO, D. D. O. **Classificação de Fluxos de Dados para Identificação de Anomalias em Honeypots com a Teoria dos Rough Sets**. RISTI (PORTO), v. E/18, p. 393-408, 2019. **QUALIS A2**.

GONÇALVES EVANGELISTA, João Rafael; DE OLIVEIRA GATTO, Dacyr Dante; SASSI, Renato José. **Classification of Web History Tools Through Web Analysis**. Lecture Notes in Computer Science, v. 11594, p. 266-276, 2019.

EVANGELISTA, J. R. G.; SILVA, E. M. L. ; SASSI, R. J. . **Enriquecimento de Base de Dorks Com Processamento de Linguagem Natural**. In: II Simpósio de Engenharia, Gestão e Inovação, 2019, Águas de Lindóia. Anais do Simpósio de Engenharia, Gestão e Inovação, 2019.

EVANGELISTA, J. R. G.; GATTO, D. D. O. ; SASSI, R. J. . **Classificação por Ranqueamento de Acesso: Análise Web em Ferramentas de Inteligência de Fontes Abertas**. In: CONBREPRO - VIII Congresso Brasileiro de Engenharia de Produção, 2018, Ponta Grossa - PR. Classificação por Ranqueamento de Acesso: Análise Web em Ferramentas de Inteligência de Fontes Abertas, 2018.

EVANGELISTA, J. R. G.; GATTO, D. D. O. ; SASSI, R. J. . **Classificação por Ranqueamento de Acesso: Análise Web em Mecanismos de Busca**. In: Seminário em Tecnologia da Informação Inteligente - SETII, 2018, São Paulo. Classificação por Ranqueamento de Acesso: Análise Web em Mecanismos de Busca, 2018. p. 98-105.

MOURA, M. L. A. O. ; J. R. G. Evangelista ; GATTO, D. D. O. ; SASSI, R. J. . **Processamento de Linguagem Natural Aplicado no Google Hacking Database**. In: 21º Simpósio de Iniciação Científica e Tecnológica (SICT), 2019, São Paulo. Boletim Técnico 48 / Simpósio de Iniciação Científica e Tecnológica - FATEC-SP. São Paulo: FATEC-SP, 2019. v. 21. p. 118-118.

MOURA, M. L. A. O. ; J. R. G. Evangelista ; GATTO, D. D. O. ; SASSI, R. J. . **Extração de Conhecimento no Google Hacking Database com Mapas Auto-Organizáveis de Kohonen**. In: XVI Encontro de Iniciação Científica - Uninove, 2019, São Paulo. XVI Encontro de Iniciação Científica, 2019. São Paulo, 2019. v. 16. p. 244-244.

EVANGELISTA, J. R. G.; SASSI, R. J. . **Classificação de Ferramentas de Histórico Web por Meio de Análise Web**. In: XV Encontro de Iniciação Científica - Universidade Nove de Julho, 2018, 2018, São Paulo. Classificação de Ferramentas de Histórico Web por Meio de Análise Web, 2018. p. 243-243.

5.2 PRÊMIOS E TÍTULOS

Menção Honrosa - XVI Encontro de Iniciação Científica da Universidade Nove de Julho – UNINOVE pelo trabalho **Extração de Conhecimento no Google Hacking Database com Mapas Auto-Organizáveis de Kohonen**. 2019.

REFERÊNCIAS

ABDELHALIM, Amany; TRAORE, Issa. The impact of google hacking on identity and application fraud. In: **2007 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing**. IEEE, p. 240-244. 2007.

ABIODUN, Oludare Isaac; JANTAN, Aman; OMOLARA, Abiodun Esther; DADA, Kemi Victoria; MOHAMED, Nachaat AbdElatif; ARSHAD, Humaira. **State-of-the-art in artificial neural network applications: A survey**. Heliyon, v. 4, n. 11, p. e00938, 2018.

ABNT – Associação Brasileira de Normas Técnicas. NBR ISO/IEC 27002. **Tecnologia da informação: Técnicas de segurança: Código de prática para a gestão da segurança da informação**. 1.ed. 2013.

ALHASSAN, J. K.; MISRA; Sanjay; UMAR, A.; MASKELIŪNAS, Rytis; DAMAŠEVIČIUS, Robertas; ADEWUMI, Adewole. **A Fuzzy Classifier-Based Penetration Testing for Web Applications**. Advances in Intelligent Systems and Computing, v. 721. p. 95-104, 2018.

ALMUBAIRIK, Norah Ahmed; WILLS, Gary. **Automated penetration testing based on a threat model**. In: Internet Technology and Secured Transactions (ICITST), 2016 11th International Conference for. IEEE, p. 413-414. 2016.

AMINI, Morteza; JALILI, Rasool; SHAHRIARI, Hamid Reza. **RT-UNNID: A practical solution to real-time network-based intrusion detection using unsupervised neural networks**. Computers & security, v. 25, n. 6, p. 459-468, 2006.

BAE, Mi Young; LIM, Han Kyu; CHO, Dae Jea. **A study on security diagnosis using automated Google hacking tools-focusing on the US government website**. Journal of Advances in Information Technology, v. 7, n. 2, p. 93-97, 2016.

BANKOVIC, Zorana *et al*. Improving security in WMNs with reputation systems and self-organizing maps. **Journal of Network and Computer Applications**, v. 34, n. 2, p. 455-463, 2011.

BAO, Wang; LIANJU, Ning; YUE, Kong. **Integration of unsupervised and supervised machine learning algorithms for credit risk assessment.** Expert Systems with Applications, v. 128, p. 301-315, 2019.

BELLA, MA Bihina; ELOFF, Jan HP. **A near-miss management system architecture for the forensic investigation of software failures**. Forensic science international, v. 259, p. 234-245, 2016.

BELLEGARDE, M.; ORVIS, M.; HELBA, S. **Ethical Hacking and Countermeasures: Attack Phases**. EC Council Press. 2010.

BERTOGLIO, Daniel Dalalana; ZORZO, Avelino Francisco. **Overview and open issues on penetration test**. Journal of the Brazilian Computer Society, v. 23, n. 1, p. 2, 2017.

CAMBAZOGLU, B. Barla; KARACA, Evren; KUCUKYILMAZ, Tayfun; TURK, Ata; AYKANAT, Cevdet. **Architecture of a grid-enabled Web search engine**. Information processing & management, v. 43, n. 3, p. 609-623, 2007.

CANTU-ORTIZ, Francisco Javier. **Advancing artificial intelligence research and dissemination through conference series: Benchmark, scientific impact and the MICAI experience**. Expert Systems with Applications, v. 41, n. 3, p. 781-785, 2014.

CARRIÓN, Joe; PUNTES, Daniel Franco; LUQUE, Emilio. **Simulating a search engine service focusing on network performance**. Procedia Computer Science, v. 108, p. 79-88. 2017.

CHAO, Chih-Yang; CHANG, Tsai-Chu; WU, Hui-Chun; LIN, Yong-Shun; CHEN, Po-Chen. **The interrelationship between intelligent agents' characteristics and users' intention in a search engine by making beliefs and perceived risks mediators**. Computers in Human Behavior, v. 64, p. 117-125. 2016.

CHEN, Mei; DÉCARY, Michel. **A Cognitive-Based Semantic Approach to Deep Content Analysis in Search Engines.** In: 2018 IEEE 12th International Conference on Semantic Computing (ICSC). IEEE, p. 131-139, 2018.

CHU, Ge; LISITSA, Alexei. Poster: **Agent-based (BDI) modeling for automation of penetration testing.** In: 2018 16th Annual Conference on Privacy, Security and Trust (PST). IEEE, p. 1-2. 2018.

CISCO. **Relatório Anual de Segurança Cibernética - Cisco 2018**. Resumo Executivo. 1.ed. 2018.

CLANCY, T. Charles; KHAWAR, Awais. Security threats to signal classifiers using self-organizing maps. In: **2009 4th International Conference on Cognitive Radio Oriented Wireless Networks and Communications**. IEEE. p. 1-6. 2009.

CLARKE, Craig. S. **Open Source Intelligence. An oxymoron or real intelligence?** Marine Corps Gazette – Professional Journal of U.S. Marines. v.99, Issue 8, p. 22, 2015.

CORCHADO, Emilio; HERRERO, Álvaro. **Neural visualization of network traffic data for intrusion detection**. Applied Soft Computing, v. 11, n. 2, p. 2042-2056, 2011.

CUI, Jie; WANG, Mingjun; LUO, Yonglong; ZHONG, Hong. **DDoS detection and defense mechanism based on cognitive-inspired computing in SDN**. Future Generation Computer Systems, v. 97, p. 275-283, 2019.

DA SILVA, Rosana Cordovil; EVANGELISTA, João Rafael Gonçalves; SASSI, Renato José; LIMA, Anderson Silva; BARBOSA, Rui Presley Duarte; GATTO, Dacyr Dante de Oliveira. **Classificação de Fluxos de Dados para Identificação de Anomalias em Honeypots com a Teoria dos Rough Sets.** Revista Ibérica de Sistemas e Tecnologias de Informação, n. E18, p. 393-408, 2019.

DEPARTAMENTO DO EXÉRCITO DOS ESTADOS UNIDOS. **"Open-Source Intelligence"**. 2012.

DEULKAR, Khushali; NARVEKAR, Meera. **An improved memetic algorithm for web search**. Procedia Computer Science, v. 45, p. 52-59, 2015.

DOBROVOLJC, Andrej; TRČEK, Denis; LIKAR, Borut. **Predicting Exploitations of Information Systems Vulnerabilities Through Attackers' Characteristics.** IEEE Access, p. 26063-26075, 2017.

EDWARDS, Matthew; LARSON, Robert; GREEN, Benjamim; RASHID, Awais; BARON, Alistair. **Panning for gold: Automatically analysing online social engineering attack surfaces**. Computers & Security, v. 69, p. 18-34, 2017.

FAN, Youping; LI, Jingjiao; ZHANG, Dai. **A Method for Identifying Critical Elements of a Cyber-Physical System Under Data Attack**. IEEE Access, v. 6, p. 16972-16984, 2018.

FENG, Nan; CHEN, Yufan; FENG, Haiyang; LI, Dahui; LI, Minqiang. **To Outsource or Not: The Impact of Information Leakage Risk on Information Security Strategy**. Information & Management, p. 103215, 2019.

FERREIRA, Ricardo Pinto; MARTINIANO, Andréa; NAPOLITANO, Domingos; ROMERO, Márcio; GATTO, Dacyr Dante de Oliveira; FARIAS; Edquel Bueno Prado; SASSI, Renato José. **Artificial Neural Network for Websites Classification with Phishing Characteristics**. Social Networking, v. 7, p. 97-109, 2018.

FERREIRA, Ricardo Pinto. **Inteligência Computacional Na Previsão Do Absenteísmo e Identificação De Tendências Absenteístas**. Tese de Doutorado. Universidade Nove de Julho. 2019.

GABINETE DE SEGURANÇA INSTITUCIONAL DA PRESIDÊNCIA DA REPÚBLICA DO BRASIL - Departamento de Segurança da Informação. **Principais dispositivos legais de caráter federal que se relacionam com Segurança da Informação**. Disponível em: < http://dsic.planalto.gov.br/documentos/quadro_legislacao.htm >. Acesso em: 07. Abril. 2019.

GALINDO, Javier Pastor, NESPOLI, Pantaleone, MARMOL, Felix Gómez, PÉREZ, Gregorio Martínez. **OSINT is the next Internet goldmine: Spain as an unexplored territory**. In Actas de las V Jornadas Nacionales de Ciberseguridad Junio 5-7, 2019, Cáceres, p. 102-109. 2019.

GERHARDT, Tatiana Engel; SILVEIRA, Denise Tolfo. **Métodos de pesquisa**. Plageder, 2009.

GIL, Antônio Carlos. **Métodos e técnicas de pesquisa social.** 6. ed. Editora Atlas SA, 2008.

GLASSMAN, Michael; KANG, Min Ju. **Intelligence in the internet age: The emergence and evolution of Open Source Intelligence (OSINT).** Computers in Human Behavior, v. 28, n. 2, p. 673-682, 2012.

GONG, S., CHO, J., LEE, C. **A Reliability Comparison Method for OSINT Validity Analysis.** IEEE Transactions on Industrial Informatics, v. 14, n. 12, p. 5428-5435. 2018.

GUO, Hui; HUANG, Shu-Guang; PAN, Zu-Lie; HU, Jian-Ping; HU, Ming-Lei. **Research on Key Data Structure Localization Technology of Buffer Overflow Vulnerability**. In: Proceedings of the 2018 International Conference on Information Science and System. p. 81-85. 2018.

HAQAF, Husam; KOYUNCU, Murat. **Understanding key skills for information security managers.** International Journal of Information Management, v. 43, p. 165-172, 2018.

HAUFE, Knut; COLOMO-PALACIOS, Ricardo; DZOMBETA, Srdan; BRANDIS, Knud; STANTCHEV, Vladimir. **Security management standards: a mapping.** Procedia Computer Science, v. 100, p. 755-761, 2016.

HATFIELD, Joseph M. **Social engineering in cybersecurity: The evolution of a concept**. Computers & Security, v. 73, p. 102-113, 2018.

HATFIELD, Joseph M. **Virtuous human hacking: The ethics of social engineering in penetration-testing.** Computers & Security, v. 83, p. 354-366, 2019.

HAYES, D. R., CAPPA, F. **Open-source intelligence for risk assessment**. Business Horizons, v. 61, n. 5, p. 689-697. 2018.

HAYKIN, Simon. **Neural Networks: A Comprehensive Foundation**. New York: Willey & Sons, 1994.

HAYKIN, Simon. **Redes Neurais - Princípios e Práticas**. 2nd Edition, Bookman, Porto Alegre. 2001.

HERZOG, P. **OSSTMM 3 –The Open Source Security Testing Methodology Manual**. ISEMCOM, 2010.

HEIDARI, Mahsa; SHAMSI, Hossein. **Analog programmable neuron and case study on VLSI implementation of MultiLayer Perceptron (MLP)**. Microelectronics Journal, v. 84, p. 36-47, 2019.

HOWELLS, Karen; ERTUGAN, Ahmet. **Applying fuzzy logic for sentiment analysis of social media network data in marketing.** Procedia Computer Science, v. 120, p. 664-670. 2017.

ISACA. (2016). **State of cybersecurity: Implications for 2016**. Disponível em: <https:// www.isaca.org/cyber/Documents/state-of-cybersecurity_res_eng_0316.pdf.>. Acesso em: 04. Dezembro. 2019.

ISO, ABNT NBR. IEC 27001: 2006: **Tecnologia da informação–Técnicas de segurança–Sistemas de gestão de segurança da informação–Requisitos.** Rio de Janeiro: ABNT, 2006.

ISO, ABNT NBR. IEC 17799: 2005: **Tecnologia da informação–Técnicas de segurança–Código de prática para a gestão da segurança da informação**. Rio de Janeiro: ABNT, 2006.

ISO/IEC. 27000: **Gestão da Segurança da Informação**. 2016 standard; 2016.

ISSAF, **Information Systems Security Assessment Framework**. Draft 0.2 B, v. 1, 2006.

JARSKE, J. M.; SEABRA, A. G.; SILVA, L. **A. Self-Organizing Maps Applied as Analysis Tool of Reading Cognitive Test**. IEEE Latin America Transactions, v.16, n.6, p.1817-1824, 2018.

KALECH, Meir. **Cyber-attack detection in SCADA systems using temporal pattern recognition techniques**. Computers & Security, v. 84, p. 225-238, 2019.

KANAKARIS, V.; TZOVELEKIS, K.; BANDEKAS, D. V. **Impact of AnonStalk (Anonymous Stalking) on users of Social Media: A Case Study**. Journal of Engineering Science & Technology Review, v. 11, n. 2, 2018.

KANGAS, Jari; KOHONEN, Teuvo; LAAKSONEN, Jorma. **Variants of self-organizing maps**. IEEE transactions on neural networks, v. 1, n. 1, p. 93-99, 1990.

KITCHENHAM, B., CHARTES, S. **Guidelines for performing Systematic Literature Reviews in Software Engineering**. Staffordshire: Elsevier, 2007.

KNOWLES, William; BARON, Alistair; MCGARR, Tim. **The simulated security assessment ecosystem: Does penetration testing need standardisation?** Computers & Security, v. 62, p. 296-316, 2016.

KOHONEN, Teuvo. Self-Organized formation of topogically correct feature maps, Biological Cybernetics, v. 43, p. 59-69. 1982.

KOHONEN, Teuvo. **Self-Organization and Associative Memory**, Series in Information Sciences, vol. 8, Springer-Verlag, Heidelberg, 1984.

KOHONEN Teuvo. **Self-organization and associative memory: 3rd ed**. New York, NY, USA: Springer-Verlag New York, Inc. 1989.

KOHONEN, Teuvo. **The self-organizing map**. Proceedings of the IEEE, v. 78, n. 9, p. 1464-1480, 1990.

KOHONEN, Teuvo. **Exploration of very large databases by self-organizing maps**. In: Proceedings of international conference on neural networks (icnn'97). IEEE, p. PL1-PL6 vol. 1. 1997.

KOHONEN, Teuvo; SOMERVUO, Panu. Self-organizing maps of symbol *strings*. **Neurocomputing**, v. 21, n. 1-3, p. 19-30, 1998.

KOHONEN, Teuvo. **Self-Organizing Maps**. Springer Series In Information Sciences. Third Edition. 2001.

KOHONEN, Teuvo. **Essentials of the self-organizing map**. Neural networks, v. 37, p. 52-65, 2013.

KOOPS, Bert-Jaap; HOEPMAN, Jaap-Henk; LEENES, Ronald. **Open-source intelligence and privacy by design.** Computer Law & Security Review, v. 29, n.6, p. 676-688, 2013.

KOTHIA, Anis; SWAR, Bobby; JAAFAR, Fehmi. **Knowledge Extraction and Integration for Information Gathering in Penetration Testing**. In: 2019 IEEE 19th International Conference on Software Quality, Reliability and Security Companion (QRS-C). IEEE, p. 330-335. 2019.

KROMBHOLZ, K., HOBEL, H., HUBER, M., WEIPPL, E. **Advanced social engineering attacks**. Journal of Information Security and applications, v. 22, p. 113-122. 2015.

LESZCZYNA, Rafał. **Standards on cyber security assessment of smart grid**. International Journal of Critical Infrastructure Protection, v. 22, p. 70-89, 2018.

LEE, Seungmin; KIM, Gisung; KIM, Sehun. **Self-adaptive and dynamic clustering for online anomaly detection**. Expert Systems with Applications, v. 38, n. 12, p. 14891-14898, 2011.

LEE, Seokcheol; SHON, Taeshik. **Open source intelligence base cyber threat inspection framework for critical infrastructures**. In: Future Technologies Conference (FTC). p. 1030-1033. 2016.

LI, Ke; WEN, Hui; LI, Hong; ZHU, Hongsong; SUN, Limin. **Security OSIF: Toward Automatic Discovery and Analysis of Event Based Cyber Threat Intelligence.** In: IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI). IEEE, 2018. p. 741-747. 2018.

LI, Ding *et al*. DDoS intrusion detection using generalized grey self-organizing maps. In: **2007 IEEE International Conference on Grey Systems and Intelligent Services**. IEEE, p. 1548-1551. 2007.

LÓPEZ, Alberto Urueña; MATEO, Fernando; NAVÍO-MARCO, Julio; MARTÍNEZ-MARTÍNEZ, José María; GOMES-SANCHÍS, Juan; VILA-FRANCÉS, Joan; SERRANO-

LÓPEZ, Antonio José. **Analysis of computer user behavior, security incidents and fraud using self-organizing maps**. Computers & Security, v. 83, p. 38-51, 2019.

LY, Pham Thi Minh; LAI, Wen-Hsiang; HSU, Chiung-Wen; SHIH, Fang-Yin. **Fuzzy AHP analysis of Internet of Things (IoT) in enterprises**. Technological Forecasting and Social Change, v. 136, p. 1-13, 2018.

MACIOŁEK, P., DOBROWOLSKI, G. **Cluo: Web-scale text mining system for open source intelligence purposes**. Computer Science, v. 14, n. 1, p. 45-62. 2013.

MANSFIELD-DEVINE, Steve. **Taking responsibility for security.** Computer Fraud & Security, v. 2015, n. 12, p. 15-18, 2015.

MATHEWS, Alex. **What can machine learning do for information security?.** Network Security, v. 2019, n. 4, p. 15-17, 2019.

MCGUFFEE, James W.; HANEBUTTE, Nadine. **Google hacking as a general education tool**. Journal of Computing Sciences in Colleges, v. 28, n. 4, p. 81-85, 2013.

MCKINNEL, Dean Richard; DARGAHI, Tooska; DEHGHANTANHA, Ali; CHOO, Kim Kwang Raymond. **A systematic literature review and meta-analysis on artificial intelligence in penetration testing and vulnerability assessment.** Computers & Electrical Engineering, v. 75, p. 175-188, 2019.

MEDKOVA, JANA. **Composition attack against social network data**. Computers & Security, v. 74, p. 115-129, 2018.

MEUCCI, Matteo; MULLER, Andrew. **The OWASP testing guide 4.0**. Open Web Application Security Project, v. 30, 2014.

MIDER, Daniel; GARLICKI, Jan; MINCEWICZ, Wojciech. **The Internet Data Collection with the Google Hacking Tool–White, Grey or Black Open-Source Intelligence?.** Przegląd Bezpieczeństwa Wewnętrznego, v. 11, n. 20, 2019.

MILLER, Daniel Bradford, GLISSON, William Bradley, YAMPOLSKIY, Mark, CHOO, Kim-Kwang Raymond. **Identifying 3D printer residual data via open-source documentation**. Computers & Security, 75, 10-23. 2018.

MITCHELL, Tom. **Machine Learning**. McGraw-Hill, 1997.

MUNIR, Rashid; MUFTI, Muhammad Rafiq; AWAN, Irfan; HU, Yim Fun; DISSO, Jules Pagna. **Detection, mit-igation and quantitative security risk assessment of invisible attacks at enterprise network**. In: 2015 3rd International Conference on Future Internet of Things and Cloud. IEEE, p. 256-263. 2015.

NAARTTIJÄRVI, Markus. **Balancing data protection and privacy–The case of information security sensor systems**. Computer Law & Security Review, v. 34, p. 1019-1038. 2018.

NAGPURE, Sangeeta; KURKURE, Sonal. **Vulnerability Assessment and Penetration Testing of Web Application**. In: 2017 International Conference on Computing, Communication, Control and Automation (ICCUBEA). IEEE, p. 1-6. 2017.

NAVARRO, Julio; DERUYVER, Aline; PARREND, Pierre. **A systematic survey on multi-step attack detection**. Computers & Security, v. 76, p. 214-249, 2018.

NAVEEN, Gouda; NAIDU, M. Ashish; RAO, B. Thirumala; RADHA, K. **A Comparative Study on Artificial Intelligence and Expert Systems**. International Research Journal of Engineering and Technology (IRJET). p. 1980-1986. 2019.

NOUBOURS, Sandra; PRITZKAU, Albert; SCHADE, Ulrich. **NLP as an essential ingredient of effective OSINT frameworks**. In: Military Communications and Information Systems Conference (MCC), IEEE. p. 1-7. 2014.

OFFENSIVE SECURITY, **Info Sec Training and Penetration Test**. Disponível em: < https://www.offensive-security.com/ >. Acesso em 11. Junho. 2019.

OTAN, Organização do Tratado do Atlântico Norte. **"NATO - The Open Source Intelligence Handbook"**. 2001.

PAN, Daoxin; BAI, Wei; ZHANG, Siyu; ZOU, Futai. **Detecting Malicious Queries from Search Engine Traf-fic.** In: 2012 8th International Conference on Wireless Communications, Networking and Mobile Computing. IEEE, p. 1-4. 2012.

PELLET, Hector; SHIAELES, Stavros; STAVROU, Stavros. **Localising social network users and profiling their movement**. Computers & Security, v. 81, p. 49-57, 2019.

POUCHARD, Line C.; DOBSON, Jonathan D.; TRIEN, Joseph P. **A Framework for the Systematic Collection of Open Source Intelligence**. In: AAAI Spring Symposium: Technosocial Predictive Analytics. p. 102-107. 2009

PTES TECHNICAL GUIDELINE. ***Pentest*-Standart.Org. – The Penetration Testing Execution Standart - 2014.** Disponível em: <http://www.*Pentest*-standard.org/index.php/PTES_Technical_Guidelines>. Acesso em: 15. Outubro. 2019.

QUICK, Darren; CHOO, Kim-Kwang Raymond. **Digital forensic intelligence: Data subsets and Open Source Intelligence (DFINT+ OSINT): A timely and cohesive mix**. Future Generation Computer Systems, v. 78, p. 558-567, 2018.

RICO, Ricardo Andrés Pinto; MEDINA, Martin José Hernández; HERNÁNDEZ, Cristian Camilo Pinzón; LÓPEZ, Daniel Orlando Díaz; RUÍZ, Juan Carlos Camilo García. **Inteligencia de fuentes abierta (OSINT) para operaciones de ciberseguridad." Aplicación de OSINT en un contexto colombiano y análisis de sentimientos"**. Revista Vínculos, v. 15, n. 2, 2018.

ROY, Ahana; MEIJA, Louis; HELLING, Paul; OLMSTED, Aspen. **Automation of cyber-reconnaissance: A Java-based open source tool for information gathering.** In: ICITST - International Conference for Internet Technology and Secured Transactions. p. 424-426. 2017.

SACHA, D., KRAUS, M., BERNARD, J., BEHRISCH, M., SCHRECK, T., ASANO, Y., & KEIM, D. A. **Somflow: Guided exploratory cluster analysis with self-organizing maps and analytic provenance**. *IEEE transactions on visualization and computer graphics*, *24*(1), 120-130. 2017.

SASSI, Renato José. **Uma arquitetura híbrida para descoberta de conhecimento em bases de dados: teoria dos Rough sets e redes neurais artificiais mapas Auto-Organizáveis**. Tese de Doutorado. Universidade de São Paulo. 2006.

SCARFONE, Karen; SOUPPAYA, Murugiah, CODY, Amanda; OREBAUGH, Angela. **NIST SP 800-115**, Technical Guide to Information Security Testing and Assessment, 2008.

SCHMIDT, Sebastian; SCHNITZER, Steffen; RENSING, Christoph. **Text classification based filters for a domain-specific search engine.** Computers in Industry, v. 78, p. 70-79, 2016.

SETTANNI, Giuseppe; SKOPIK, Florian; SHOVGENYA, Yegor; FIEDLER, Roman; CAROLAN, Mark; CONROY, Damien; BOETTINGER, Konstantin; GALL, Mark; BROST, Gerd; PONCHEL, Christophe; HAUSTEIN, Mirko; KAUFMANN, Helmut; THEUERKAUF, Klaus; OLLI, Pia. **A collaborative cyber incident management system for European interconnected critical infrastructures**. Journal of Information Security and Applications, v. 34, p. 166-182, 2017.

SIMON, Kai. **Vulnerability Analysis Using Google and Shodan.** Lecture Notes in Computer Science, v. 10052, p. 725-730, 2016.

SIRIPANADORN, Supakit; HATTAGAM, Wipawee; TEAUMROONG, Neung. **Anomaly detection in wireless sensor networks using self-organizing map and wavelets**. International Journal of Communications, v. 4, n. 3, p. 74-83, 2010.

SRINIVAS, Jangirala; DAS, Ashok Kumar; KUMAR, Neeraj. **Government regulations in cyber security: Framework, standards and recommendations**. Future Generation Computer Systems, v. 92, p. 178-188, 2019.

SRIVASTAVA, Amit Kumar; KUMAR, Shishir. **An effective computational technique for taxonomic position of security vulnerability in software development**. Journal of Computational Science, v. 25, p. 388-396, 2018.

STEFINKO, Yaroslav; PISKOZUB, Andrian; BANAKH, Roman. **Manual and automated penetration testing. Benefits and drawbacks**. Modern tendency. In: Modern Problems of Radio Engineering. Telecommunications and Computer Science (TCSET), 2016 13th International Conference on. IEEE, p. 488-491. 2016.

STEINBART, Paul John; RASCHKE, Robyn L.; GAL, Graham; DILLA, William N. **The influence of a good relationship between the internal audit and information security functions on information security outcomes.** Accounting, Organizations and Society, v. 71, p. 15-29, 2018.

SUN, Shiliang; LUO, Chen; CHEN, Junyu. **A review of natural language processing techniques for opinion mining systems**. Information fusion, v. 36, p. 10-25, 2017.

TALWAR, Rohit; KOURY, April. **Artificial intelligence–the next frontier in IT security?**. Network Security, v. 2017, n. 4, p. 14-17, 2017.

TANG, Andrew. **A guide to penetration testing**. Network Security, v. 2014, n. 8, p. 8-11, 2014.

TETSKYI, Artem; KHARCHENKO, Vyacheslav; UZUN, Dmytro. **Neural networks based choice of tools for penetration testing of web applications**. In: 2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies (DESSERT). IEEE, p. 402-405. 2018.

TOFFALINI, Flavio; ABBÀ, Maurizio; CARRA, Damiano; BALZAROTTI, Davide. **Google dorks: Analysis, creation, and new defenses**. Lecture Notes in Computer Science. v. 9721, p. 255-275. 2016.

VACAS, I., MEDEIROS, I., NEVES, N. **Detecting Network Threats using OSINT Knowledge-Based IDS**. In 2018 14th European Dependable Computing Conference (EDCC). p. 128-135. IEEE. 2018.

VENTER, Hein S.; ELOFF, Jan HP; LI, Y. L. **Standardising vulnerability categories.** Computers & Security, v. 27, n. 3-4, p. 71-83, 2008.

VIJAYAKUMAR, Biveeken; FUAD, Muhammad Marwan Muhammad. **A New Method to Identify Short-Text Authors Using Combinations of Machine Learning and Natural Language Processing Techniques**. Procedia Computer Science, v. 159, p. 428-436, 2019.

VIJAYAKUMAR, S.; SHESHADRI, K. N. **Applications of Artificial Intelligence in Academic Libraries**. International Journal of Computer Sciences and Engineering. v. 7, p. 136-140. 2019.

VISCOVERY.NET, **Viscovery SOMine 5**. Disponível em: https:// https://www.viscovery.net/. Acesso em: 15 de Janeiro de 2020.

WATTERS, P. A.; LAYTON, R. **Automating Open Source Intelligence: Algorithms for OSINT**. USA. Syngress. 2016.

WANG, Meng; LU, Yiqin; QIN, Jiancheng. **A dynamic MLP-based DDoS attack detection method using feature selection and feedback**. Computers & Security, v. 88, p. 101645, 2020.

WEISHÄUPL, Eva; YASASIN, Emrah; SCHRYEN, Guido. **Information security investments: An exploratory multiple case study on decision-making, evaluation and learning**. Computers & Security, v. 77, p. 807-823, 2018.

YOU, Wei; ZONG, Peiyuan; CHEN, Kai; WANG, XiaoFeng; LIAO, Xiaojing; BIAN, Pan. **Semfuzz: Semantics-based automatic generation of proof-of-concept exploits**. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. 2017. p. 2139-2154.

ZHANG, Jialong; NOTANI, Jayant; GU, Guofei. **Characterizing google hacking: A first large-scale quantitative study.** Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering. Springer, v. 152, p. 602-622, 2015.

ZHAO, Shi-Wei; CAO, Ze-Wen; LIU, Wen-Sen. OSIA: **Open Source Intelligence Analysis System Based on Cloud Computing and Domestic Platform**. In: 2nd International Conference on Information Science and Control Engineering. IEEE, 2015. p. 371-375. 2015.

ZHAO, Zhongwen; DAI, Yingchun. **A new method of vulnerability taxonomy based on information security attributes**. In: 2012 IEEE 12th International Conference on Computer and Information Technology. IEEE, p. 739-741. 2012.

ZEROUAL, Imad; LAKHOUAJA, Abdelhak. **Data science in light of natural language processing: An overview.** Procedia Computer Science, v. 127, p. 82-91, 2018.

APÊNDICE A – DISPOSITIVOS LEGAIS DE CARÁTER FEDERAL QUE SE RELACIONAM COM SEGURANÇA DA INFORMAÇÃO

Neste apêndice são descritos os dispositivos legais de caráter federal do Departamento de Segurança da Informação, que abordam o tema segurança da informação. O departamento pertence ao Gabinete de Segurança Institucional da Presidência da República do Brasil.

Dispositivo Legal	Mandamento Legal / Punição	Segurança da Informação
Constituição Federal, art. 5º, inciso X.	Direito à privacidade.	Sigilo das informações relacionadas à intimidade ou à vida privada de alguém.
Constituição Federal, art. 5º, inciso XII.	Direito à privacidade das comunicações.	Sigilo dos dados telemáticos e das comunicações privadas.
Código Penal, art. 153, § 1º-A.	Pena de 1 a 4 anos e multa por crime de divulgação de documento confidencial contido ou não nos sistemas ou bancos de dados da Administração Pública.	Proteção do sigilo das informações classificadas constantes nos sistemas ou bancos de dados da Administração Pública.
Código Penal, art. 154.	Pena de 3 meses a um ano, ou multa por crime de violação de segredo profissional.	Proteção do sigilo das informações conhecidas em razão de função, ministério, ofício ou profissão.
Código Penal, art. 184, § 3º.	Pena de 2 a 4 anos por crime de violação de direito autoral mediante cabo, fibra ótica, satélite, ondas ou qualquer outro sistema.	Proteção da autenticidade.
Código Penal, art. 266, § 1º e 2º.	Pena - detenção, de 1 mês a 1 ano, ou multa	Proteção a não interrupção ou perturbação de serviço telegráfico, telefônico, informático, telemático ou de informação de utilidade pública.
Código Penal, art. 305.	Pena de 2 a 6 anos e multa por crime de supressão, destruição ou ocultação de documento público ou particular.	Proteção da disponibilidade e integridade das informações constantes nos órgãos e entidades públicos.
Código Penal, art. 313-A.	Pena de 2 a 12 anos e multa por crime de inserção de dados falsos em sistema informatizado ou banco de dados da Administração Pública, alteração ou exclusão de dados corretos.	Proteção da integridade e disponibilidade das informações constantes nos órgãos e entidades públicos.
Código Penal, art. 313-B.	Pena de 3 meses a 2 anos e multa por crime de modificação ou alteração não autorizada de sistemas de informações.	Proteção da integridade e disponibilidade das informações constantes nos órgãos e entidades públicos.
Lei nº 7.170/83, art. 13.	Pena de 3 a 15 anos por crime espionagem ou divulgação de informações sigilosas a grupo estrangeiro, ou a organização ou grupo de existência ilegal.	Proteção das informações sigilosas relacionadas à segurança nacional.
Lei nº 9.279/96, art. 195, inciso XI.	Constitui crime de concorrência desleal divulgar, explorar ou utilizar, sem autorização, de conhecimentos, informações ou dados confidenciais, utilizáveis na indústria, comércio ou prestação de serviços, excluídos aqueles que sejam de conhecimento público ou que sejam evidentes para um técnico no assunto, a que teve acesso mediante relação contratual ou empregatícia,	Proteção da privacidade das pessoas jurídicas, relacionado ao sigilo de suas informações.

Dispositivo Legal	Mandamento Legal / Punição	Segurança da Informação
	mesmo após o término do contrato.	
Lei nº 9.296/96, art. 10.	Pena de dois a quatro anos, e multa por crime de interceptação de comunicações telefônicas, de informática ou telemática, ou quebra de segredo da Justiça, sem autorização judicial ou com objetivos não autorizados em lei.	Sigilo dos dados e das comunicações privadas.
Lei nº 9.472/97, art. 3º, inciso V.	O usuário de serviços de telecomunicações tem direito à inviolabilidade e ao segredo de sua comunicação, salvo nas hipóteses e condições constitucional e legalmente previstas.	Sigilo das comunicações.
Lei nº 9.472/97, art. 3º, inciso VI.	O usuário de serviços de telecomunicações tem direito à não divulgação, caso o requeira, de seu código de acesso.	Proteção de informações pessoais de caráter sigiloso.
Lei nº 9.472/97, art. 3º, inciso IX.	O usuário de serviços de telecomunicações tem direito ao respeito de sua privacidade nos documentos de cobrança e na utilização de seus dados pessoais pela prestadora do serviço.	Proteção de informações pessoais de caráter sigiloso.
Lei nº 9.605/98, art. 62.	Pena de 1 a 3 anos e multa pela conduta de destruir, inutilizar ou deteriorar arquivo, registro, museu, biblioteca, pinacoteca, instalação científica ou similar protegido por lei, ato administrativo ou decisão judicial.	Disponibilidade e integridade de dados e informações.
Decreto nº 3.505/00, art. 1º.	Institui a Política de Segurança da Informação nos órgãos e entidades da Administração Pública Federal.	Pressupostos básicos da segurança da informação.
Lei nº 10.683/03, art. 6º, inciso IV.	Prevê a competência do GSIPR de coordenar a atividade de segurança da informação.	Todos os aspectos da segurança da informação.
Lei n. º 12.737/12, de 30 de novembro de 2012. Lei "Carolina Dieckmann"	Dispõe sobre a tipificação criminal de delitos informáticos; altera o Decreto-Lei nº 2.848, de 7 de dezembro de 1940 - Código Penal; e dá outras providências. - Invasão de dispositivo informático - Interrupção ou perturbação de serviço telegráfico, telefônico, informático, telemático ou de informação de utilidade pública - Falsificação de documento particular - Falsificação de cartão	Todos os aspectos da segurança da informação.
Lei nº 12.965, de 23 abril de 2014. (Marco Civil da Internet)	Estabelece princípios, garantias, direitos e deveres para o uso da Internet no Brasil.	Segurança jurídica para os usuários da rede, sejam eles usuários, empresas, provedores e Administração Pública.

APÊNDICE B – PUBLICAÇÕES QUE ABORDAM OSINT COM IA

Neste apêndice são descritas todas as publicações encontradas na revisão sistemática da literatura que abordam OSINT com Inteligência Artificial.

Título da Publicação	**Área de Aplicação**	**Autores**	**Ano**
A dynamic games approach to proactive defense strategies against Advanced Persistent Threats in cyber-physical systems	Cybersecurity	HUANG, Linan; ZHU, Quanyan	2020
An investigation of using classification techniques in prediction of type of targets in Cyber attacks	Cybersecurity	Sina Pournouri, Shahrzad Zargari, Babak Akhgar	2019
BlackWidow: Monitoring the Dark Web for Cyber Security Information	Cybersecurity	Matthias Schafer, Markus Fuchs, Martin Strohmeier, Markus Engel, Marc Liechti, Vincent Lenders	2019
Cognitive security: A comprehensive study of cognitive science in cybersecurity	Cybersecurity	Roberto O Andrade, Sang Guun Yoo Facultad	2019
Design of a Classification Model for a Twitter-based Streaming Threat Monitor	Cybersecurity	Fernando Alves, Pedro M. Ferreira, Alysson Bessani	2019
Developing insights from social media using semantic lexical chains to mine short text structures	Social Media	Cecil Eng Huang Chua, Veda C. Storey, Xiaolin Li, Mala Kaul	2019
Enhancing Information Sharing and Visualization Capabilities in Security Data Analytic Platforms	Cybersecurity	Gustavo Gonzalez-Granadillo, Mario Faiella, Ibéria Medeiros, Rui Azevedo, Susana Gonzalez-Zarzosa	2019
Gathering Cyber Threat Intelligence from Twitter Using Novelty Classification	Cybersecurity	Le, B. D., Wang, G., Nasim, M., & Babar, M. A	2019
Localising social network users and profiling their movement	Cybersecurity	Hector Pellet, Stavros Shiaeles, Stavros Stavrou	2019
PURE: Generating Quality Threat Intelligence by Clustering and Correlating OSINT	Cybersecurity	Azevedo, Rui; Medeiros, Ibéria; Bessani, Alysson.	2019
Searching for Extremist Content Online Using the Dark Crawler and Sentiment Analysis	Military Purposes	Ryan Scrivens, Tiana Gaudette, Garth Davies, Richard Frank	2019
The impact of preprocessing in natural language for open source intelligence and criminal investigation	Cybersecurity	Johnsen, Jan William; Franke, Katrin	2019
Turkish national cyber-firewall to mitigate countrywide cyber-attacks	Cybersecurity	Arif Sari	2019
A Supervised Machine Learning Based Approach for Automatically Extracting High-Level Threat Intelligence from Unstructured Sources	Cybersecurity	Yumna Ghazi, Zahid Anwar, Rafia Mumtaz, Shahzad Saleem and Ali Tahir.	2018
A survey on technical threat intelligence in the age of sophisticated cyber attacks	Cybersecurity	Wiem Tounsi, Helmi Rais	2018
Detecting Network Threats using OSINT Knowledge-based IDS	Cybersecurity	Ivo Vacas, Ibéria Medeiros, Nuno Neves	2018

Título da Publicação	**Área de Aplicação**	**Autores**	**Ano**
Evaluating Automated Facial Age Estimation Techniques for Digital Forensics	Cybersecurity	Felix Anda, David Lillis, Nhien-An Le-Khac, Mark Scanlon	2018
Impact of AnonStalk (Anonymous Stalking) on users of Social Media: a Case Study	Cybersecurity	V. Kanakaris, K. Tzovelekis and D. V. Bandekas	2018
Is quantum computing becoming relevant to cyber-security?	Cybersecurity	Keegan Keplinger	2018
Managing cyber threat intelligence in a graph database	Cybersecurity	Seulgi Lee, Hyeisun Cho, Nakhyun Kim, Byungik Kim, Junhyung Park	2018
Modeling The Causes Of Terrorism From Media News: An Innovative Framework Connecting Impactful Events With Terror Incidents	Military Purposes	Truong Son Pham, Tuan-Hao Hoang	2018
Ontology population for open-source intelligence: A GATE-based solution	Languages and Translations	Giulio Ganino, Domenico Lembo, Massimo Mecella, Federico Scafoglieri	2018
Open source intelligence (OSINT) as support of cybersecurity operations. “Use of OSINT in a colombian context and sentiment Analysis”	Cybersecurity	Ricardo Andrés Pinto Rico, Martin José Hernández Medina, Cristian Camilo Pinzón Hernández, Daniel Orlando Díaz López, Juan Carlos Camilo García Ruíz	2018
Security OSIF: Toward Automatic Discovery and Analysis of Event Based Cyber Threat Intelligence	Cybersecurity	Ke Li, Hui Wen, Hong Li, Hongsong Zhu, Limin Sun	2018
Using Deep Neural Networks to Translate Multi-lingual Threat Intelligence	Languages and Translations	Priyanka Ranade, Sudip Mittal, Anupam Joshi and Karuna Joshi	2018
Applying fuzzy logic for sentiment analysis of social media network data in marketing	Social Media	Karen Howellsa, Ahmet Ertugan	2017
Classification of Colloquial Arabic Tweets in real- time to detect high-risk floods	Languages and Translations	Waleed Alabbas, Haider M. al-Khateeb, Ali Mansour, Gregory Epiphaniou, Ingo Frommholz	2017
Cloud security issues and challenges: a survey	Cybersecurity	Ashish Singh, Kakali Chatterjee	2017
Extracting Cyber Threat Intelligence From Hacker Forums: Support Vector Machines versus Convolutional Neural Networks	Cybersecurity	Isuf Deliu, Carl Leichter, Katrin Franke	2017
Towards a breakthrough Speaker Identification approach for Law Enforcement Agencies: SIIP	Languages and Translations	Khaled Khelif, Yann Mombrun, Gerhard Backfried, Farhan Sahito, Luca Scarpato, Petr Motlicek, Srikanth Madikeri, Damien Kelly, Gideon Hazzani, Emmanouil Chatzigavriil	2017
Utility and potential of rapid epidemic intelligence from internet-based sources	Social Media	S.J. Yan, A.A. Chughtai, C.R. Macintyre	2017

Título da Publicação	**Área de Aplicação**	**Autores**	**Ano**
A problem shared is a problem halved: a survey on the dimensions of collective cyber defense through security information sharing	Cybersecurity	Florian Skopik, Giuseppe Settanni, Roman Fiedler	2016
Automating social network analysis: A power tool for counter-terrorism	Military Purposes	Leslie Ball	2016
Building Document Treatment Chains Using Reinforcement Learning and Intuitive Feedback	Business and Industry	Esther Nicart, Bruno Zanuttini, Hugo Gilbert, Bruno Grilhères, Frederic Praca	2016
How to Apply Privacy by Design in OSINT and big Data Analytics?	Cybersecurity	Jyri Rajamäki, Jussi Simola	2016
Sampling Labelled Profile Data for Identity Resolution	Social Media	Matthew Edwards, Stephen Wattam, Paul Rayson and Awais Rashid	2016
A Systematic Survey of Online Data Mining Technology Intended for Law Enforcement	Cybersecurity	Matthew Edwards, Awais Rashid, And Paul Rayson	2015
Social Opinion Mining : an approach for Italian language	Social Media	Vito Santarcangelo, Giuseppe Oddo, Maria Pilato, Fabrizio Valenti, Claudio Fornaro	2015
CAPER: Crawling and Analysing Facebook for Intelligence Purposes	Social Media	Carlo Aliprandi, Antonio E. De Luca, Giulia Di Pietro, Matteo Raffaelli, Davide Gazzè, Mariantonietta N. La Polla, Andrea Marchetti, Maurizio Tesconi	2014
Crawling Open-source Data for Indicators of Human Trafficking	Military Purposes	Ben Brewster, Timothy Ingle, Glynn Rankin	2014
Foraging Online Social Networks	Social Media	Gijs Koot, Mirjam A.A. Huis in 't Veld, Joost Hendricksen, Rianne Kaptein, Arnout de Vries, Egon L. van den Broek	2014
NLP as an Essential Ingredient of Effective OSINT Frameworks	OSINT	Sandra Noubours, Albert Pritzkau, Ulrich Schade	2014
OSINT for B2B platforms	Business and Industry	V.F. Pais, D.S. Ciobanu	2014
Semantic Crawling: an Approach based on Named Entity Recognition	Cybersecurity	Giulia Di Pietro, Carlo Aliprandi, Antonio E. De Luca, Matteo Raffaelli, Tiziana Soru	2014
The Big Data Imperative Air - Force Intelligence for the Information Age	Military Purposes	Col Shane P. Hamilton, Lt Col Michael P. Kreuzer	2014
TheGame: an evaluation on Self Organization & Engagement by semantic analysis	Languages and Translations	Giovanna Ferrari, Nicoletta Magnetti, Paolo Marianib, Federico Neri	2014
Can we trust this user? Predicting insider's attitude via YouTube usage profiling	Social Media	Miltiadis Kandias, Vasilis Stavrou, Nick Bozovic, Lilian Mitrou, Dimitris Gritzalis	2013
Cluo: Web-Scale Text Mining System For Open Source Intelligence Purposes	Languages and Translations	Przemysław Maciołek, Grzegorz Dobrowolski	2013

Título da Publicação	**Área de Aplicação**	**Autores**	**Ano**
Massively Scalable Near Duplicate Detection in Streams of Documents using MDSH	Web Texts and Documents	Paul Logasa Bogen II, Christopher T. Symons, Amber McKenzie, Robert M. Patton, Robert E. Gillen	2013
Proactive Insider Threat Detection Through Social Media: The YouTube Case	Social Media	Miltiadis Kandias, Vasilis Stavrou, Nick Bozovic, Dimitris Gritzalis	2013
Automatic Exploitation of Multilingual Information for Military Intelligence Purposes	Military Purposes	Sandra Noubours, Matthias Hecking	2012
Hybrid model of content extraction	Web Texts and Documents	Pir Abdul Rasool Qureshi, Nasrullah Memon	2012
Challenges in Open Source Intelligence	Military Purposes	Clive Best	2011
Data mining with LinkedIn	Cybersecurity	Danny Bradbury	2011
LanguageNet: A Novel Frameworlk for Processing Unstructured Text Information	Web Texts and Documents	Pir Abdul Rasool Qureshi , Nasrullah Memon, Uffe Kock Wiil	2011
Desktop Text Mining for Law Enforcement	Languages and Translations	Jonathan Brett Crawley, Gerhard Wagner	2010
Detecting Terrorism Evidence in Text Documents	Military Purposes	Pir Abdul Rasool Qureshi, Nasrullah Memon, Uffe Kock Wiil	2010
Using Term Extraction Patterns to Discover Coherent Relationships from Open Source Intelligence	Cybersecurity	William L. Sousan, Qiuming Zhu, Robin Gandhi, William Mahoney, Anup Sharma	2010
WISDOM from Light-Weight Information Retrieval	Social Media	David B. Bracewell, Steven Gustafson, Abha Moitra and Gregg Steuben	2010
Near Real Time Information Mining in Multilingual News	Languages and Translations	M. Atkinson, E. Van der Goot	2009
Web Mining for Open Source Intelligence	Languages and Translations	Clive Best	2008
Toward an interoperable dynamic network analysis toolkit	Social Media	Kathleen M. Carley, Jana Diesner, Jeffrey Reminga, Maksim Tsvetovat	2007
Textually Retrieved Event Analysis Toolset	Military Purposes	John Palmer	2005

APÊNDICE C – OUTLIERS ENCONTRADOS NA BASE DE DORKS

Neste apêndice são descritos todos os Outliers encontrados na fase E da abordagem proposta neste trabalho. Encontrou-se o total de 102 Outliers, sendo que 93 foram tratados e 9 foram removidos.

	Outliers Tratados
	Outliers Removidos

intitle:"TRENDnet" (inurl:"top.htm"| inurl:"STSSYS.HTM"| inurl:"AVIEW.HTM"| inurl:"JPlug.htm" | inurl:"JVIEW.HTM")
inurl:"home.htm?cat=home" | inurl:"index.htm?cat=info" | inurl:"index.htm?cat=settings" | inurl:"index.htm?cat=network" | inurl:"index.htm?cat=bluetooth"
inurl:"info_deviceStatus.html" | inurl:"info_suppliesStatus.html" | inurl:"info_configuration.html" | inurl:"info_config_network.html" | inurl:"info_specialPages.html" | inurl:"info_colorUsageJobLog.html" | inurl:"info_eventLog.html"
intext:VIEWS · Server: - Database: information_schema - Table: SCHEMA_PRIVILEGES · Browse · Structure · SQL · Search · Export
http://www.google.com/search?source=ig&hl=fr&rlz=&q=allinurl:+Category.php%3FIndustrYID%3D
phpLDAPadmin intitle:phpLDAPadmin filetype:php inurl:tree.php | inurl:login.php | inurl:donate.php (0.9.6 | 0.9.7)
"Powered by: Land Down Under 800" | "Powered by: Land Down Under 801" - www.neocrome.net
intitle:"WEB//NEWS Personal Newsmanagement" intext:"Ãƒâ€šÃ‚Â© 2002-2004 by Christian Scheb - Stylemotion.de"+"Version 1.4 "+"Login"
"CosmoShop by Zaunz Publishing" inurl:"cgi-bin/cosmoshop/lshop.cgi" -johnny.ihackstuff.com -V8.10.106 -V8.10.100 -V.8.10.85 -V8.10.108 -V8.11*
("Skin Design by Amie of Intense")|("Fanfiction Categories" "Featured Stories")|("default2, 3column, Romance, eFiction")
inurl:docmgr | intitle:"DocMGR" "enter your Username and"|"und Passwort bitte"|"saisir votre nom"|"su nombre de usuario" -ext:pdf -inurl:"download.php
"PhpCollab . Log In" | "NetOffice . Log In" | (intitle:"index.of." intitle:phpcollab|netoffice inurl:phpcollab|netoffice -gentoo)
"Powered by PHP-Fusion v6.00.110" | "Powered by PHP-Fusion v6.00.2.." | "Powered by PHP-Fusion v6.00.3.." -v6.00.400 -johnny.ihackstuff
"powered by phplist" | inurl:"lists/?p=subscribe" | inurl:"lists/index.php?p=subscribe" -ubbi -bugs +phplist -tincan.co.uk
"toendaCMS is Free Software released under the GNU/GPL License." | "powered by toendaCMS" -inurl:demo
intext:"Free Ecommerce Shopping Cart Software by ViArt" +"Your shopping cart is empty!" + "Products Search" +"Advanced Search" + "All Categories"
intext:"RPG Inferno is not available to guests" or intext:"Battle Ground Â· Clans Â· Store Â· Jobs Â· Auction Â· Spells Shop Â· Statistics Â· Member List"
http://www.google.com/#sclient=psy&hl=en&safe=off&site=&source=hp&q=:inurl%3Amj_wwwusr&aq=f&aqi=&aql=&oq=&pbx=1&fp=2dcb6979649afcb0
inurl:"jscripts/tiny_mce/plugins/tinybrowser/" OR inurl:"jscripts/tiny_mce/plugins/tinybrowser/" "index of"

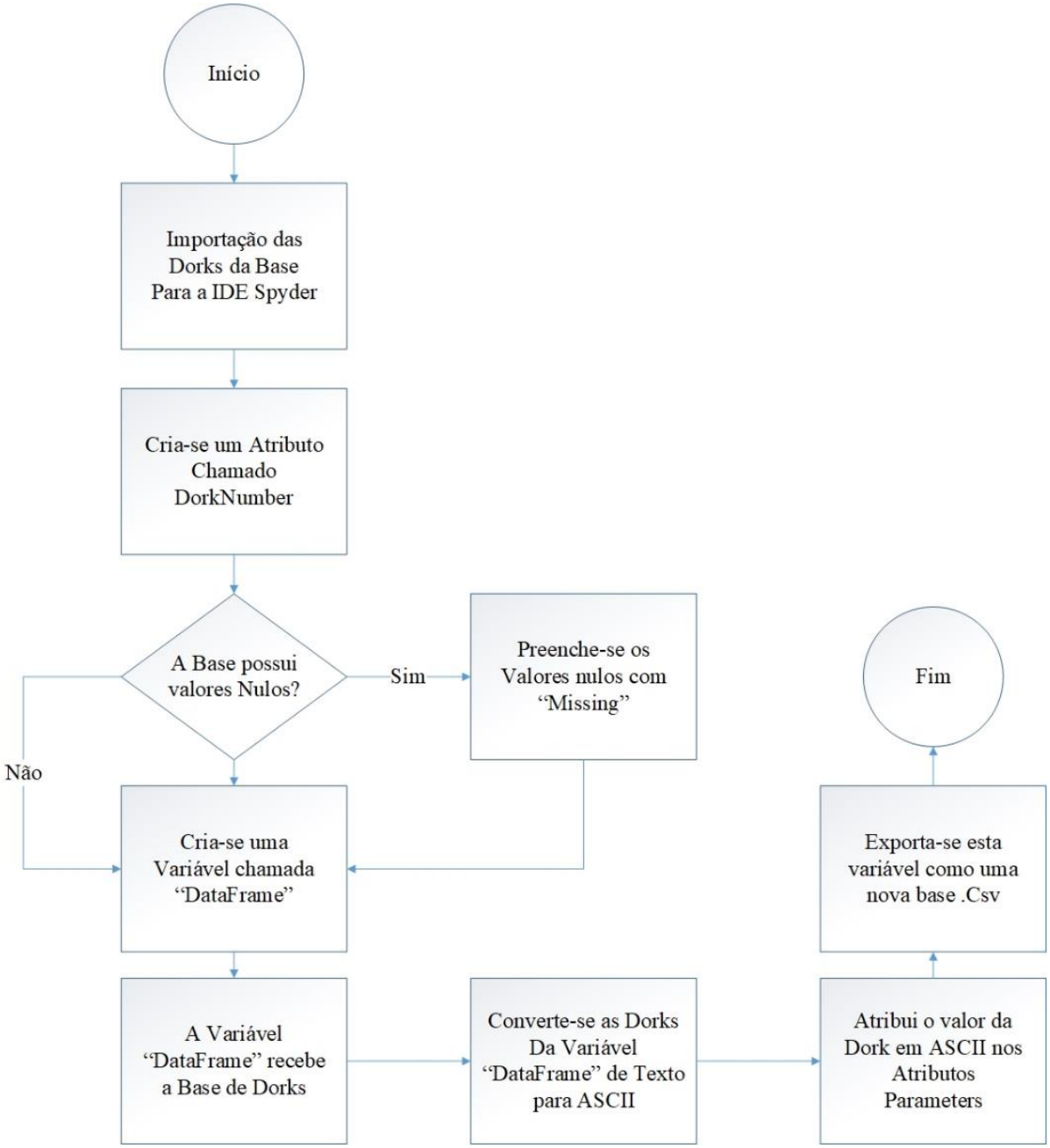
("Fiery WebTools" inurl:index2.html) | "WebTools enable * * observe, *, * * * flow * print jobs"
intitle:"Network Print Server" filetype:shtm (inurl:u_printjobs | inurl:u_server | inurl:a_server | inurl:u_generalhelp | u_printjobs)
intext:"Please enter correct password for Administrator Access. Thank you" "Copyright Ãƒâ€šÃ‚Â© 2003 SMC Networks, Inc. All rights reserved."
intitle:"Flash Operator Panel" -ext:php -wiki -cms -inurl:asternic -inurl:sip -intitle:ANNOUNCE -inurl:lists
http://www.google.com/search?q=intitle:%22Network+Storage+Link+for+USB+2.0+Disks%22+Firmware&num=100&hl=en&lr=&c2coff=1&safe=off&filter=0
intitle:Cisco "You are using an old browser or have disabled javascript. You must use version 4 or higher of Netscape Navigator/Communicator"
intitle:"Live View / - AXIS" | inurl:view/view.shtml OR inurl:view/indexFrame.shtml | intitle:"MJPG Live Demo" | "intext:Select preset position"
allintitle: Axis 2.10 OR 2.12 OR 2.30 OR 2.31 OR 2.32 "Network Camera "
inurl:RgFirewallRL.asp | inurl:RgDmzHost.asp | inurl:RgMacFiltering.asp | inurl:RgConnect.asp | inurl:RgEventLog.asp | inurl:RgSecurity.asp | inurl:RgContentFilter.asp | inurl:wlanRadio.asp
intext:VIEWS · Server: - Database: information_schema · Browse · Structure · SQL · Search · Export
intitle:"eMule *" intitle:"- Web Control Panel" intext:"Web Control Panel" "Enter your password here."
Login ("Powered by Jetbox One CMS ÃƒÂ¢Ã¢â‚¬Å¾Ã‚Â¢" | "Powered by Jetstream Ãƒâ€šÃ‚© *")
intitle:"Member Login" "NOTE: Your browser must have cookies enabled in order to log into the site." ext:php OR ext:cgi
intitle:"Cisco CallManager User Options Log On" "Please enter your User ID and Password in the spaces provided below and click the Log On button to co
intitle:"Content Management System" "user name"|"password"|"admin" "Microsoft IE 5.5" -mambo -johnny.ihackstuff
intitle:"Content Management System" "user name"|"password"|"admin" "Microsoft IE 5.5" -mambo -johnny.ihackstuff
intitle:"Login Forum Powered By AnyBoard" intitle:"If you are a new user:" intext:"Forum Powered By AnyBoard" inurl:gochat -edu
intitle:"*- HP WBEM Login" | "You are being prompted to provide login account information for *" | "Please provide the information requested and press
intext:"Fill out the form below completely to change your password and user name. If new username is left blank, your old one will be assumed." -edu
intitle:("TrackerCam Application Login") -trackercam.com
intitle:"mikrotik routeros > administration" intext:"mikrotik routeros" intext:"configuration page" -demo intext:"Mikrotik, RouterOS and the Mikrotik logo are registered trademarks of Mikrotikls SIA"
"Help * Contact * Imprint * Sitemap" | "powered by papoo" | "powered by cms papoo"
intitle:"mikrotik routeros > administration" intext:"Mikrotik, RouterOS and the Mikrotik logo are registered trademarks of Mikrotikls SIA"
intitle:"Openbravo" (inurl:"openbravo/security/Login_FS.html" | inurl:"openbravo/security/Login_Welcome.html" | inurl:"openbravo/security/Login_F1.html" | inurl:"openbravo/security/Login_F0.html")
intitle:"Honeywell XL Web Controller - Login" (inurl:"standard/default.php" | inurl:"standard/header/header.php" | inurl:"standard/mainframe.php" | inurl:"standard/footer/footer.php" | inurl:"standard/update.php")
-english -help -printing -companies -archive -wizard -pastebin -adult -keywords "Warning: this page requires Javascript. To correctly view, please enable it in your browser"
inurl:"NmConsole/Login.asp" intitle:"Login - Ipswitch WhatsUp Professional 2005" "Ipswitch, Inc"

inurl:"/munin/network-*.html" OR inurl:"/munin/apache-*.html" OR inurl:"/munin/disk-*.html" OR inurl:"/munin/system-*.html" OR inurl:"/munin/munin-*.html" OR inurl:"/munin/problems.html"
(username=* | username:* |) | (((password=* | password:*) | (passwd=* | passwd:*) | (credentials=* | credentials:*)) | ((hash=* | hash:*) | (md5:* | md5=*)) | (inurl:auth | inurl:passwd | inurl:pass)) filetype:log
inurl:"passes" OR inurl:"passwords" OR inurl:"credentials" -search -download -techsupt -git -games -gz -bypass -exe filetype:txt @yahoo.com OR @gmail OR @hotmail OR @rediff
inurl:"ftp" intext:"user" | "username" | "userID" | "user ID" | "logon" | "login" intext:"password" | "passcode" filetype:xls | filetype:xlsx
ext:(doc | pdf | xls | txt | ps | rtf | odt | sxw | psw | ppt | pps | xml) (intext:confidential salary | intext:"budget approved") inurl:confidential
(intitle:WebStatistica inurl:main.php) | (intitle:"WebSTATISTICA server") -inurl:statsoft -inurl:statsoftsa -inurl:statsoftinc.com -edu -software -rob
Google Dork inurl:Curriculum Vitale filetype:doc (Vital Informaticon , Addres, Telephone Numer, SSN , Full Name, Work , etc) In Spanish.
Google Dork For Social Security Number (In Spain and Argentina is D.N.I)
"OpenSSL" AND "1.0.1 Server at" OR "1.0.1a Server at" OR "1.0.1b Server at" OR "1.0.1c Server at" OR "1.0.1d Server at" OR "1.0.1e Server at" OR "1.0.1f Server at"
intitle:"ERROR: The requested URL could not be retrieved" "While trying to retrieve the URL" "The following error was encountered:"
intitle:"WSO 2.4" [Sec. Info], [Files], [Console], [Sql], [Php], [Safe mode], [*String* tools], [Bruteforce], [Network], [Self remove]
inurl:"/tiny_mce/plugins/ajaxfilemanager/inc/data.php" | inurl:"/tiny_mce/plugins/ajaxfilemanager/ajax_create_folder.php" -github
(intitle:"phpRemoteView") `rwx` "uname"
"Powered by Invision Power File Manager" (inurl:login.php) | (intitle:"Browsing directory /")
intitle:"Index Of" intext:"iCloud Photos" OR intext:"My Photo Stream" OR intext:"Camera Roll"
intext:"Powered by phpSQLiteCMS" | intitle:"phpSQLiteCMS - A simple & lightweight CMS"
inurl:"server-status" "Server Version: Apache/" "Server Built: " "Server uptime:"
intitle:"Document title goes here" intitle:"used by web search tools" " example of a simple Home Page"
"Novell, Inc" WEBACCESS Username Password "Version *.*" Copyright -inurl:help -guides|guide
filetype:pl -intext:"/usr/bin/perl" inurl:webcal (inurl:webcal | inurl:add | inurl:delete | inurl:config)
inurl:CHANGELOG.txt intext:drupal intext:"SA-CORE" -intext:7.32 -site:github.com -site:drupal.org
MySQL: ON MSSQL: OFF Oracle: OFF MSSQL: OFF PostgreSQL: OFF cURL: ON WGet: ON Fetch: OFF Perl: ON
intitle:"RouterOS" intitle:"configuration page" intext:"You have connected to a router. Administrative access only."
inurl:"*.php?*=*.php" intext:"Warning: include" -inurl:.html -site:"php.net" -site:"stackoverflow.com" -inurl:"*forums*"
"Warning:" "SAFE MODE Restriction in effect." "The script whose uid is" "is not allowed to access owned by uid 0 in" "on line"
"There seems to have been a problem with the" " Please try again by clicking the Refresh button in your web browser."
-wizard -pastebin -adult -keywords "Warning: this page requires Javascript. To correctly view, please enable it in your browser"
-english -help -printing -companies -archive"Warning: this page requires Javascript. To correctly view, please enable it in your browser"
inurl:"ftp" intext:"user" | "username" | "userID" | "login" intext:"password" | "passcode" filetype:xls | filetype:xlsx

intext:smtp | pop3 intext:login | logon intext:password | passcode filetype:xls | filetype:xlsx
intitle:"Belarc Advisor Current Profile" intext:"Click here for Belarc's PC Management products, for large and small companies."
intitle:osCommerce inurl:admin intext:"redistributable under the GNU" -demo -site:oscommerce.com
intitle:"AppServ Open Project *" "AppServ is a merging open source software installer package" -phpbb
(intitle:"PRTG Traffic Grapher" inurl:"allsensors")|(intitle:"PRTG Traffic Grapher - Monitoring Results")
intitle:".:: Welcome to the Web-Based Configurator::." & intext:"Welcome to your router Configuration Interface"
intitle:"WebMail | Powered by Winmail Server - Login" & (intext:"Username" & intext:"Password")
"Web-Based Management" "Please input password to login" -inurl:johnny.ihackstuff.com
Copyright @ 2007 Powered By Hot or Not Clone by Jnshosts.com Rate My Pic :: Home :: Advertise :: Contact us::
filetype:php inurl:index.php inurl:"module=subjects" inurl:"func=*" (listpages| viewpage | listcat)
"Powered *: newtelligence" ("dasBlog 1.6"| "dasBlog 1.5"| "dasBlog 1.4"|"dasBlog 1.3")
intitle:"Flash Operator Panel" -ext:php -wiki -cms -inurl:asternic -inurl:sip -intitle:ANNOUNCE -inurl:lists
intext:"you to handle frequent configuration jobs easily and quickly"
inurl:comment.asp intext: Battle Blog cannot notify you of these activities unless you supply an accurate e-mail.
"This WebUI administration tool requires scripting support" intitle:'Login' intext:'Admin Name:' -score
(intext:mail AND intext:samAccountName) AND (filetype:xlsx OR filetype:xls OR filetype:csv OR filetype:txt)
intext:5baa61e4c9b93f3f0682250b6cf8331b7ee68fd8 AND (ext:txt OR ext:csv OR ext:xls OR ext:lst)
ext:txt inurl:gov intext:"Content-Type: text/plain; charset=utf-8" AND intext:"Received: from "
intext:phpMyAdmin SQL Dump filetype:sql intext:INSERT INTO `admin` (`id`, `user`, `password`) VALUES -github
intitle:"Please enter your User ID and Password in the spaces provided below and click the Log On button to co
intitle:"Member Login" "NOTE: Your browser must have cookies." ext:php
intitle:"Member Login" "NOTE: Your browser must have cookies enabled in order to log into the site." ext:cgi
"You have requested access to a restricted area of our website. Please authenticate yourself to continue."
"index of /" (upload.cfm | upload.asp | upload.php | upload.cgi | upload.jsp | upload.pl)
intitle:"ERROR: The requested URL could not be retrieved" "The following error was encountered:"
ext:php intext:"$dbms""$dbhost""$dbuser""$dbpasswd""$table_prefix""phpbb_installed"
inurl:"NmConsole/Login.asp" intitle:"Login - Ipswitch WhatsUp Professional 2005" "Ipswitch, Inc"

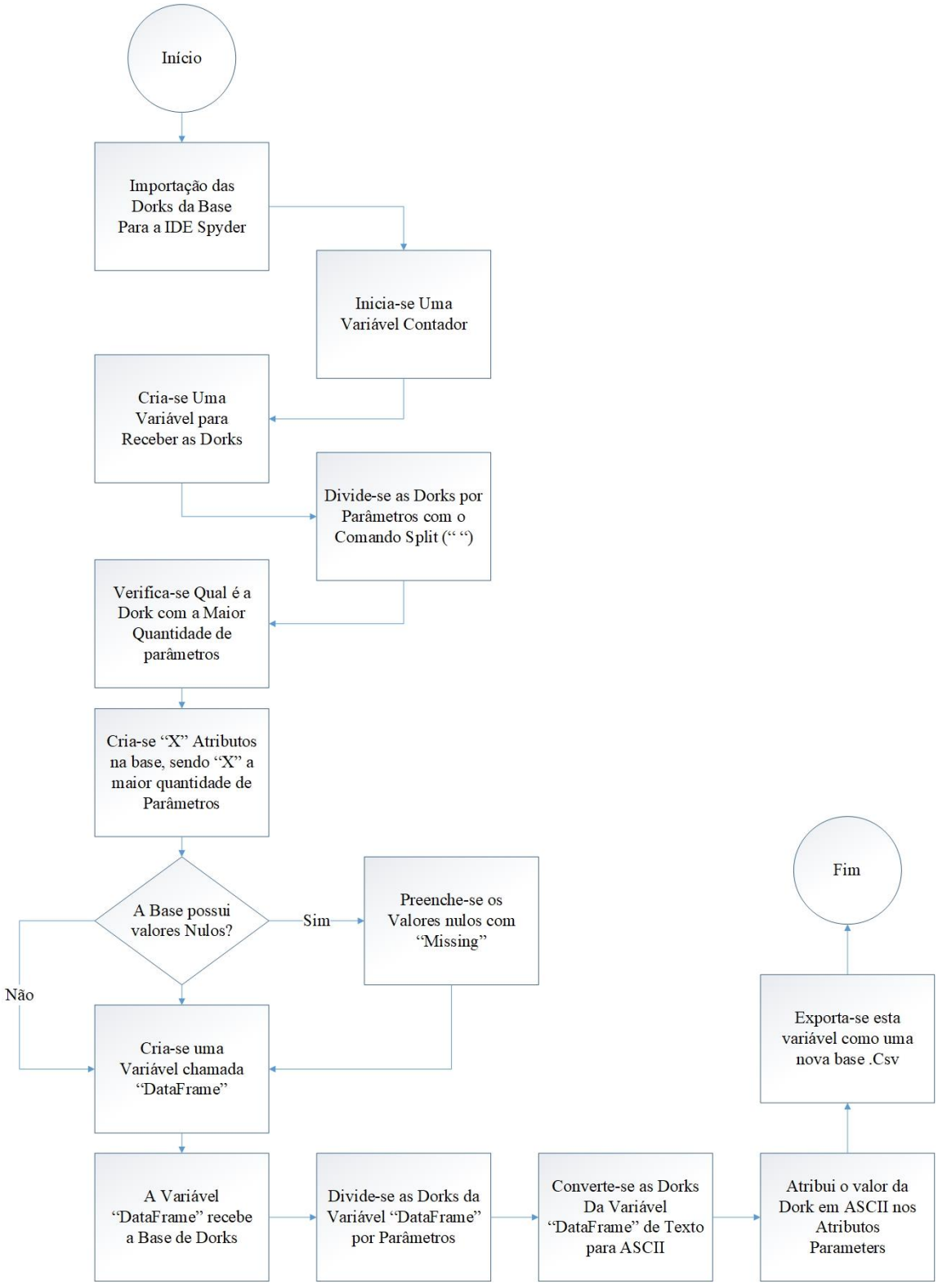
APÊNDICE D – FLUXOGRAMA PARA CONVERTER DORKS EM ASCII

Neste apêndice é apresentado o fluxograma para converter os caracteres da Dork em ASCII.



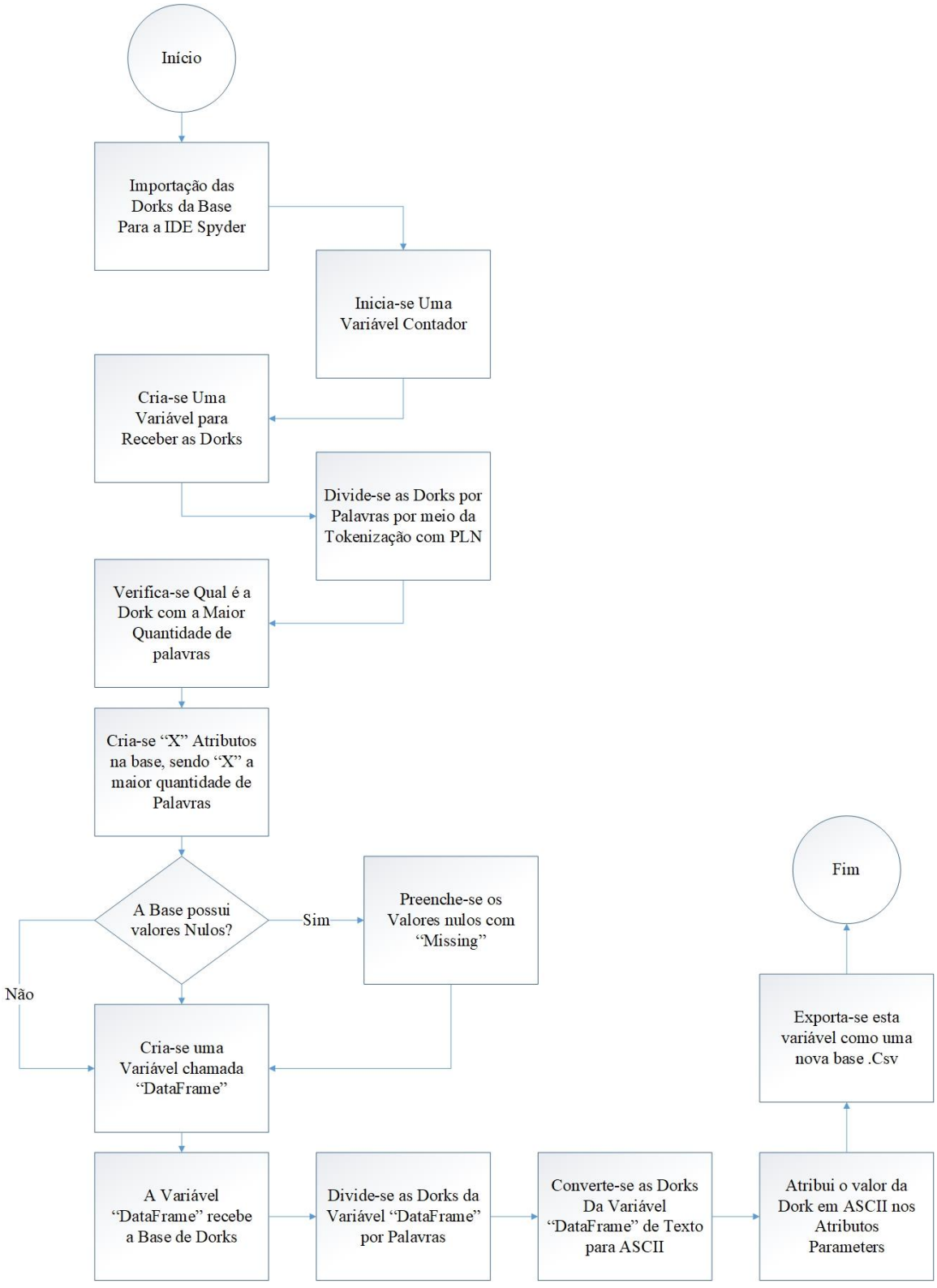
APÊNDICE E – FLUXOGRAMA PARA DIVIDIR DORKS POR PARÂMETROS E CONVERTER PARA ASCII

Neste apêndice é apresentado o fluxograma para dividir a Dork por parâmetros e converter seus caracteres em ASCII.



APÊNDICE F – FLUXOGRAMA PARA DIVIDIR DORKS POR PALAVRAS E CONVERTER PARA ASCII

Neste apêndice é apresentado o fluxograma para dividir a Dork por palavras e converter seus caracteres em ASCII.



APÊNDICE G – FLUXOGRAMA PARA DIVIDIR DORKS POR CARACTERES E CONVERTER PARA ASCII

Neste apêndice é apresentado o fluxograma para dividir a Dork por caracteres e converter seus caracteres em ASCII.