NGAV (Next-Generation Antivirus) Especialista na Detecção de *Cyber*-Ataques

Sidney M. L. Lima [1], Ricardo P. Pinheiro [2], Danilo M. Souza [2], Sthéfano H. M. T. Silva [2], Petrônio G. Lopes [2], Rafael D. T. de Lima [2], Jemerson R. de Oliveira [2], Thyago de A. Monteiro [2], Sérgio M. M. Fernandes [2], Edison de Q. Albuquerque [2]

[1] Departamento de Eletrônica e Sistemas, Universidade Federal de Pernambuco – Recife, Brasil

[2] Departamento de Computação, Universidade de Pernambuco – Recife, Brasil

sidney.lima@ufpe.br,{rpp3, dms2, shtms, pgl, rdtl, jro, tam, smurilo, edison}@ecomp.poli.br

Abstract. *Almost all malware running on web-server are PHP codes. The present paper creates a NGAV (Next-Generation Antivirus) expert in auditing web-based threats, specifically from PHP files, in real time. Our antivirus monitors 11,777 malicious behaviors that cyber-attacks can do when executing directly from a malicious web server to a service on a personal computer. Our NGAV achieves an average accuracy of 97.50% in distinguishing between benign and malware web scripts through data science, artificial intelligence and machine learning. Our antivirus can supply the limitations of the commercial antiviruses as for the detection of Web fileless attack.*

Keywords*. Malware; Antivirus; Artificial Neural Networks; Real-time malware detection; Computer Forensics.*

Resumo. *Quase todos os malware executados em servidores web são códigos PHP. Este trabalho cria um NGAV (Next Generation Antivirus - Antivírus de Próxima Geração) especialista em detectar ameaças web em arquivos PHP, em tempo real. Nosso antivírus monitora 11.777 comportamentos maliciosos que o cyber-ataque possa fazer quando executado diretamente de um servidor web malicioso para um serviço em um computador pessoal. O nosso NGAV alcança uma precisão média de 97.50% na distinção entre scripts web benignos e malware através de ciência dos dados, inteligência artificial e máquinas de aprendizado estatístico. O nosso antivírus tem a capacidade de suprir as limitações do antivírus comerciais quanto à detecção de cyber-ataques oriundos de servidores web.*

Palavras-Chave. *Malware; Antivírus; Redes Neurais Artificiais; Detecção de Malware em tempo real; Forense computacional.*

1. Introdução

A internet vem se caracterizando como o principal meio de comunicação na sociedade contemporânea. A internet se notabiliza pela convergência de todos os meios de comunicação previamente existentes. Através da rede mundial de computadores, é possível assistir televisão, ouvir rádio, ler jornal e ter acesso a qualquer outro forma de transmissão de informação entre diferentes povos, idiomas e culturas. Com a popularização da internet, os estudantes criam seu próprio ambiente virtual de estudo,

proveem seu próprio conteúdo e interagem de forma ativa e constante na busca pelo conhecimento. A rede mundial de computadores impulsiona a criatividade, habilidades tecnológicas, possibilita a abertura de distintas visões, além de habilidades de comunicação e de aprendizado (MINNESOTA/USA, 2008).

Como efeito colateral, a crescente popularização da internet propicia que a produção de malware continue crescendo de forma rápida ainda durante alguns anos visto que a internet é o principal meio de propagação de aplicações maliciosas (INTEL, 2018). Apenas em 2016, foram lançados mais de 7.100.000 (sete milhões e cem mil) malware, um aumento de 47,3% em relação ao ano de 2015 (INTEL, 2018). "Malware" é uma junção dos termos "malicioso" e "software". O malware tem como principal objetivo acessar um dispositivo alheio sem permissão explícita de seu proprietário (LIMA, *et al.*, 2018)(CERT.BR, 2016). Logo, senhas bancárias, redes sociais, fotos ou vídeos íntimos podem ser furtados a partir da *cyber*-infecção por malware.

Enfatiza-se que grande parte dos transtornos provocados por malware são irreversíveis. Logo, cada vez mais se vem investindo na segurança digital através de novas tecnologias em antivírus, firewall e biometria. Estima-se que os serviços de antivírus estão presentes em 95% dos computadores pessoais, além de 84% dos internautas terem serviços de firewall e 82% possuírem atualizações automáticas ativadas no seu sistema operacional Microsoft (MICROSOFT, 2017).

Apesar da presença massiva de mecanismos de *cyber*-vigilância em praticamente todos os computadores pessoais, os ataques cibernéticos vêm causando prejuízos bilionários e em escalas cada vez maiores (MICROSOFT, 2017). Uma das razões desse insucesso é porque assim que uma vulnerabilidade é solucionada, *cyber*-criminosos surgem com outra tática (SOPHOS, 2014). Atualmente, ao invés de infecções convencionais, através de arquivos executáveis portáveis, os *cyber*-ataques modernos empregam ataques "sem arquivos" ou *fileless*. Tecnicamente, ataques "sem arquivos" são lançados diretamente de um servidor web malicioso em direção a um serviço responsivo em um computador pessoal (CONRAD, *et al.*, 2017). Em 2017, das novas vulnerabilidades observadas, apenas 24% eram originadas no computador pessoal, e as demais (76%) advinham do servidor. A redução dos ataques provenientes do cliente reflete a tendência na diminuição de táticas invasivas as quais visam atingir o cliente diretamente (SKYBOX, 2018).

Pesquisa da *Symantec* estima que as principais formas de *cyber*-infecções, pela internet, são sites comuns que foram comprometidos e infectados com código malicioso (SYMANTEC, 2012). A lista completa contendo as categorias mais perigosas de ataques a páginas web pode ser vistas na Figura 1. É interessante notar que sites de conteúdo adulto/pornográfico estão fora do top cinco, em décimo lugar. Por sua vez, os blogs, comumente com conteúdo ideológico e religioso, possuem em média o triplo de ameaças em comparação a sites adulto-pornográficos (SYMANTEC, 2012). Conclui-se que, independentemente do seu comportamento, um internauta não está a salvo quanto a infecções visto que conselhos convencionais já não são mais úteis como, por exemplo, não acessar sites pornográficos visando evitar cyber-invasões.

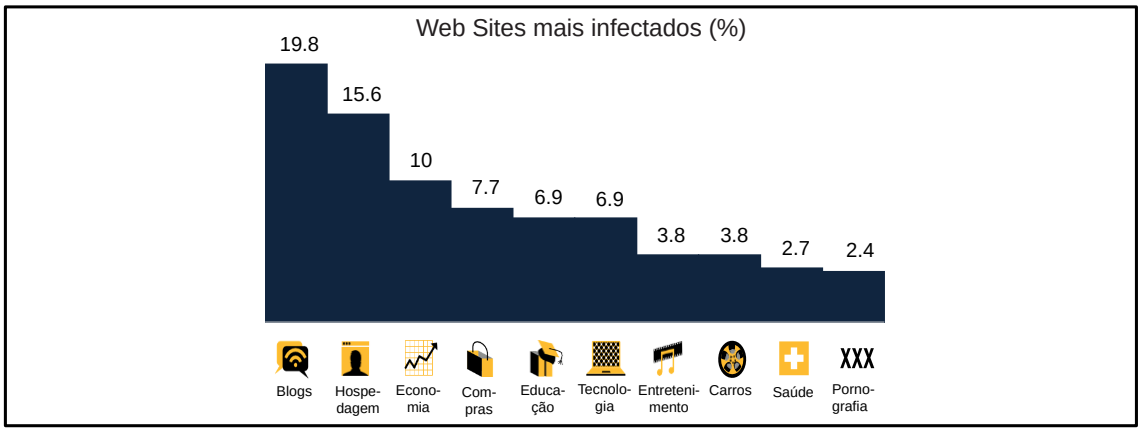

Figura 1. Categorias mais perigosas de Web Sites, de acordo com a Symantec (SYMANTEC, 2012).

Uma vez que os scripts maliciosos estejam instalados nos sites comprometidos, eles lançam dinamicamente ataques por meio de navegadores web para a vítima (computador pessoal). Quase todos os arquivos contendo malware executados em servidores web são códigos PHP (uma linguagem em scripts para servidores, comumente usada em sites) (SOPHOS, 2014). É crescente o uso de scripts PHP maliciosos confeccionados para que servidores disseminem atividades ilegais (SOPHOS, 2014). Como adversidade, ataques originados no servidor via malware PHP são bastante difíceis de catalogar (SOPHOS, 2014). Graças à natureza do negócio de hospedagem de sites, quando um uma empresa descobre que um de seus servidores está infectado, é bem mais vantajoso simplesmente apagar este servidor e configurar um novo ao invés de tentar descobrir o que aconteceu (SOPHOS, 2014). Como nem tais empresas nem seus parceiros de segurança sabem exatamente o que aconteceu, geralmente o novo servidor é rapidamente infectado também (SOPHOS, 2014).

Em síntese, ataques oriundos de servidores web, por meio de malware PHP, tem capacidade de ludibriar os mecanismos de *cyber*-vigilância (SOPHOS, 2014). Então, o trabalho proposto investiga (*i*) os principais antivírus comerciais quanto à perícia forense de PHP maliciosos. A detecção de malware variou entre 0% a 78,50% a depender do antivírus. Em média, houve a detecção 16,82% das pragas virtuais. Com aspecto desfavorável, os antivírus, em média, atestaram falsos negativos e foram omissos em 49,49% e 33,69% dos casos, respectivamente. Além disso, cerca de 60% dos antivírus não foram capazes de diagnosticar qualquer uma das amostras maliciosas. Enfatiza-se que em nosso estudo, os malware PHP analisados têm as suas atuações maliciosas documentadas e catalogadas em bancos de dados (VIRUSSHARE, 2019). Mesmo assim, mais da metade dos antivírus comerciais avaliados não tinham qualquer conhecimento sobre as existências dos arquivos PHP malware investigados.

Há uma gama de motivos para que o ataque “sem arquivos” represente um desafio tão relevante para os antivírus tradicionais (PALOALTO, 2013). Uma delas diz respeito à aquisição do arquivo PHP visto que seria necessária a permissão do provedor de conteúdo de onde ele foi executado remotamente. Na prática forense digital, no entanto, empresas de hospedagem costumam trabalhar de forma desintegrada e não compartilhar informações com as empresas de *cyber*-segurança (SOPHOS, 2014). Logo, as estratégias de atuação das empresas de hospedagem web atrapalham e retardam o enfrentamento a malware PHP.

Atualmente, as organizações buscam suprir as deficiências dos antivírus tradicionais através de mecanismos de *cyber*-segurança nomeados de NGAVs (*Next Generation Antivirus*). As soluções NGAVs buscam reconhecer padrão de comportamento de malware através do uso de inteligência artificial, aprendizagem de máquina e ciência de dados (SANS, 2019). A recomendação dos pesquisadores é que os NGAVs adicionem várias camadas de inteligência na detecção de *cyber*-ameaças (SKYCURE, 2016). Logo, os NGAVs do estado-da-arte propõem extrair características do arquivo, de maneira preventiva, antes de executá-lo. O executável passa por um processo de *disassembling* visando a Engenharia Reversa do arquivo suspeito. Logo, o executável pode ser estudado e, portanto, é possível investigar a intenção maliciosa do arquivo através de máquinas de aprendizado estatístico. Essa metodologia, nomeada de análise estática, é capaz de obter taxas médias de acertos superiores a 90% na detecção de aplicativos malware (LIMA, *et al.*, 2018).

A análise estática, no entanto, pode ser facilmente contornada por um ataque web "sem arquivos". Em síntese, análise estática de características é inválida mediante ataque "sem arquivos" visto que não há como um computador pessoal periciar códigos fontes armazenados e executados em um servidor web remoto. A incapacidade da análise estática em detectar ataques "sem arquivos" tem mudado o foco da pesquisa de malware para a determinação de características que possam identificar comportamento malicioso como um processo, e não pelos meios empregados na análise estática. Então, ao invés da impraticável análise estática, a extração de características do nosso NGAV diz respeito à auditória do comportamento do sistema operacional e não mais a inexequível análise do código fonte do arquivo suspeito.

De modo a validar o nosso NGAV, o trabalho proposto desenvolve um ambiente controlado nomeado *Web-Server Next Generation Sandbox*. Em nosso ambiente, são virtualizados o servidor web malicioso e o computador pessoal, respectivamente. Então, o computador pessoal (cliente) requisita a página Web suspeita do servidor virtualizado. A partir daí, os comportamentos maliciosos, oriundos do ataque "sem arquivos", são auditados por nossa *Web-Server Next Generation Sandbox*. Tais comportamentos maliciosos servem como atributos de entrada das máquinas de aprendizado estatístico.

Quantos aos cenários e experimentos, são explorados diferentes parâmetros das redes neurais artificiais empregadas como máquina de aprendizado estatístico. O nosso NGAV (*iii*) alcança um desempenho médio de 97.50% na distinção entre aplicativos PHP benignos e malware. Então, o presente artigo demonstra que a inteligência artificial é uma boa alternativa para as fabricantes dos antivírus comerciais. As limitações dos mecanismos de *cyber*-segurança tradicionais podem ser supridas por nosso NGAV dotado de ambiente controlado especialista em auditar ataques *fileless*. Ao invés de modelos baseados em listas negras, a nossa *engine* emprega ciência de dados, aprendizagem de máquina e inteligência artificial na identificação de comportamentos maliciosos.

O trabalho proposto tem, como destaque, as seguintes contribuições:

- Investigação dos principais 86 antivírus comerciais quanto à identificação de arquivos PHP malware. Em média, houve a detecção 16,82% das pragas virtuais.

- O nosso NGAV (*Next Generation Antivirus*) alcança um desempenho médio de 97.50% na distinção entre arquivos benignos e *malware*, acompanhado de um tempo de treinamento médio de apenas 0,04 segundos [1].
- O antivírus criado possibilita a detecção preventiva das ameaças virtuais, em ambiente controlado, antes de alcançarem as máquinas dos clientes.
- Criação de base de dados, nomeada PAEMAL, visando ser empregada, como *benchmark*, na verificação da qualidade dos antivírus.

2. Limitações dos Antivírus Comerciais

Apesar de ser questionado há mais de uma década, o *modus operandi* dos antivírus é baseado em assinaturas quando o arquivo suspeito é consultado em bases de dados nomeadas de lista negra (SANS, 2019). Isso quer dizer, o malware suspeito é comparado a uma lista negra confeccionada a partir de denúncias prévias e isso requer que alguns clientes já tenham sido infectados. Antivírus baseados em detecção de assinaturas possuem bons resultados quando se deparam com ameaças conhecidas, mas encontram grandes dificuldades no combate a aplicativos malware recém-criados (PALOALTO, 2013). Tal dificuldade é agravada em relação a códigos PHP mal-intencionados visto que navegar, assim como outras atividades web, são de tempo real por natureza (PALOALTO, 2013).

Logo, basta que o *hash* do arquivo investigado não esteja na lista negra do antivírus para que o malware não seja detectado. O *hash* funciona como um identificador único de um dado arquivo. Então, dadas as limitações dos antivírus comerciais, não é uma tarefa difícil desenvolver e distribuir variantes de uma aplicação mal intencionada. Para isso, basta fazer pequenas alterações no malware original com rotinas que, efetivamente, não tem qualquer utilidade a exemplo de laços de repetição e desvios condicionais sem instruções em seus escopos. Essas alterações sem utilidade, no entanto, tornam o *hash* do malware modificado diferente do *hash* do malware original. Consequentemente, o malware, incrementado com rotinas nulas, não será detectado pelo antivírus o qual catalogou o malware original. Cabe ressaltar a existência de ferramentas (*exploits*) responsáveis por criar e distribuir variantes, de forma automatizada, de um mesmo malware original. Conclui-se que antivírus, baseados em assinaturas, têm efetividade nula quando submetidos a variantes de um mesmo malware (SANS, 2019).

Por intermédio da plataforma VirusTotal, o trabalho proposto investiga os principais antivírus comerciais com seus respectivos resultados apresentados na Tabela 1. Os resultados dos 86 principais antivírus comerciais estão disponíveis no nosso repositório autoral (PAEMAL, 2019). Foram empregados 200 PHP maliciosos. O objetivo é verificar a quantidade de pragas virtuais catalogadas pelos antivírus. A motivação é que a aquisição de novas pragas virtuais assume papel importante no combate a aplicações mal-intencionadas. Logo, quanto maior for a base de dados de malware, nomeada de lista negra, melhor tende a ser a defesa provida pelo antivírus. A Figura 2 exibe o diagrama da metodologia proposta em diagrama de blocos. Inicialmente, os aplicativos malware são enviados ao servidor pertencente à plataforma VirusTotal

[1] Na primeira linha da Tabela 3, há a descrição técnica da melhor configuração adotada pelo nosso NGAV (*Next Generation Antivirus*).

(VIRUSTOTAL, 2019). Após isso, eles são analisados pelos antivírus comerciais vinculados ao VirusTotal. A plataforma permite a possibilidade de emissão de três tipos diferentes de diagnósticos; malware, benigno e omissão.

Quanto à primeira possibilidade do VirusTotal, o antivírus detecta a malignidade do arquivo suspeito. No ambiente experimental proposto, todos os arquivos submetidos são aplicativos malware de domínio público (VIRUS, 2019). Logo, o antivírus acerta quando detecta a malignidade do arquivo investigado. A detecção do malware indica que o antivírus provê um serviço robusto contra *cyber*-invasões. Na segunda possibilidade, o antivírus atesta a benignidade do arquivo investigado. Logo, no estudo proposto, quando o antivírus alega a benignidade do arquivo, trata-se de um caso de falso negativo visto que todas as amostras são maliciosas. Isso quer dizer, o arquivo investigado é malware, no entanto, o antivírus atesta benignidade, de forma equivocada. Na terceira possibilidade, o antivírus não emite opinião sobre o arquivo suspeito. A omissão indica que o arquivo investigado jamais foi avaliado pelo antivírus tão pouco ele possui robustez para avaliá-lo em tempo real. A omissão do diagnóstico, por parte do antivírus, aponta a sua limitação quanto a serviços em larga escala.

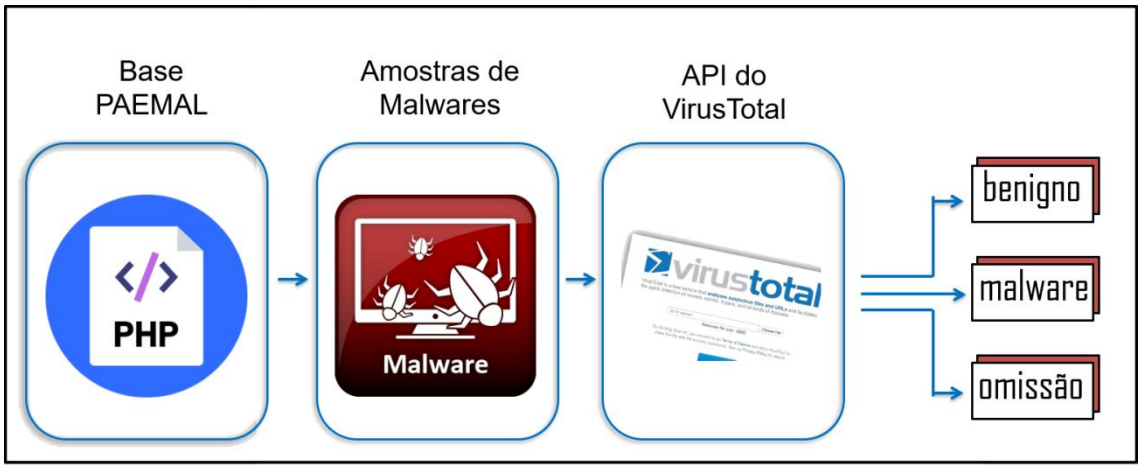

Figura 2. Diagrama da API do VirusTotal

A detecção dos aplicativos malware PHP variou de 0% a 78,50% a depender do antivírus avaliado. Em média, os 86 antivírus foram capazes de detectar 16,82% das pragas virtuais avaliadas, com desvio padrão de 21,88. O desvio padrão elevado indica que a detecção de arquivos maliciosos pode sofrer variações abruptas a depender do antivírus escolhido. Determina-se que a proteção, contra invasões cibernéticas, está em função da escolha de um antivírus robusto dotado de uma grande e atualizada lista negra. Em média, os antivírus atestaram falsos negativos em 49,49% dos casos, com desvio padrão de 38,32. Atestar a benignidade de um malware pode implicar em prejuízos irrecuperáveis. Uma pessoa ou instituição, por exemplo, passaria a confiar em uma determinada aplicação maliciosa quando, de fato, trata-se de um malware. Ainda como aspecto desfavorável, cerca de 57% não emitiram opinião em qualquer uma das 200 amostras maliciosas. Em média, os antivírus foram omissos em 33,68% dos casos, com desvio padrão de 45.61. A omissão do diagnóstico aponta a limitação dos antivírus quanto à detecção de malware em tempo real.

Tabela 1. Resultado dos melhores e piores antivírus comerciais em relação à malware PHP. Resultados expandidos estão no nosso repositório autoral (PAEMAL, 2019).

Antivírus	Detecção (%)	Falso negativo (%)	Omissão (%)
Ikarus	**78.50**	16.00	5.50
GData	59.00	39.50	1.50
AegisLab	55.50	42.50	2.00
Avast	54.50	45.50	0.00
MAX	54.50	42.50	3.00
AVG	54.00	46,00	0.00
Kaspersky	50.50	48.00	1.50
ZoneAlarm	50.50	48,00	1.50
Avira	49.00	50.00	1.00
MicroWorld-eScan	47.00	53.00	0.00
BitDefender	47.00	51.50	1.50
Ad-Aware	47.00	50.50	2.50
Emsisoft	47.00	53.00	0.00
ALYac	46.50	52.00	1.50
F-Prot	5.00	94.50	0.50
TrendMicro	4.00	93.00	3.00
ClamAV	3.50	94.50	2.00
VIPRE	3.50	96.00	0.50
TotalDefense	2.50	97.00	0.50
Jiangmin	2.50	96.00	1.50
AhnLab-V3	2.00	98.00	0.00
K7GW	1.50	98.50	0.00
K7AntiVirus	1.50	98.50	0.00
VBA32	1.00	98.50	0.50
nProtect	0.50	44.50	55.00
ViRobot	0.50	99.50	0.00
Yandex	0.50	98.50	1.00
Panda	0.50	99.50	0.00

Tabela 2: Miscelânea de classificações providas pelos antivírus comerciais. O total de classificações está no repositório autoral (PAEMAL, 2019).

Antivírus	VirusShare_f0cba054906c17c94b6852c6088b47b0.php	VirusShare_13f71688e77649255460d68258f3e450.php	VirusShare_d293667cf4aad8acce2b86258462bcdb.php
Ikarus	Trojan.JS.Tadtruss	Backdoor.IRCBot.ADDS	JS:Trojan.JS.Agent.SJP
GData	Benign	Win.Trojan.Downloader-68	Benign
AegisLab	Trojan.Script.Agent.dtkph	Benign	Benign
Avast	Benign	Suspicious_GEN.F47V0405	Suspicious_GEN.F47V0614
MAX	EXP/Blacole.EB.4	malware	malware
AVG	JS:Decode-DB	PHP:Multicom-A	JS:Agent-DWO
Kaspersky	Win.Trojan.Iframe-68	Backdoor.IRCBot.ADDS	JS:Trojan.JS.Agent.SJP
ZoneAlarm	Benign	Benign	Benign
Avira	Benign	Benign	Benign
MicroWorld-eScan	Benign	Benign	JS.TrojanjQuery.8C8B
BitDefender	Trojan.JS.Iframe.wq	Benign	HEUR:Trojan.Script.Generic
Ad-Aware	Benign	Benign	Benign
Emsisoft	Trojan.JS.IFrame.ANM	Benign	HTML/Phishing.m
ALYac	JS/BlacoleRef.E	JS.Redirector.AX	Benign
Baidu	Trojan.JS.IFrame.ANM	Benign	Benign
Bkav	Omission	Omission	Omission
McAfee-GW-Edition	HEUR_HTJS.HDJSFN	Benign	Benign
Arcabit	Exploit	TrojanDownloader:PHP/RunShell.A	Benign
McAfee	JS.Blacole.H	PHP/SillyDlScript.HFI	Benign
Antiy-AVL	Malware	PHP/Downloader.A	HTML/Infected.WebPage.Gen2
F-Secure	TrojWare.JS.Agent.exi	Benign	TrojWare.JS.Agent.CUS

Inclui-se como adversidade, no combate a aplicações mal intencionadas, o fato dos antivírus comerciais não possuírem um padrão na classificação dos aplicativos *malware* como visto na Tabela 2. Nós escolhemos 3 dos 998 arquivos malware PHP de modo a exemplificar a miscelânea de classificações dadas pelos antivírus comerciais. Como não existe um padrão, os antivírus dão os nomes que desejam, por exemplo, o McAfee-GW-Edition pode identificar um *malware* como "HEUR_HTJS.HDJSFN" e o McAfee, pertencente a mesma empresa, identificá-lo como "JS.Blacole.H". Logo, a falta de um padrão atrapalha as estratégias de *cyber*-vigilância visto que cada categoria de malware deve ter tratamentos (vacinas) distintos. Conclui-se que é inviável o aprendizado de máquina supervisionado visando reconhecimento de padrão de categorias de PHP malware. Devido a esse emaranhado confuso de classificação multi-classe, providas pelos

especialistas (antivírus) como visto na Tabela 2, é estatisticamente improvável que alguma técnica de aprendizado de máquina adquira capacidade de generalização.

3. Estado-da-Arte

O *modus operandi* dos antivírus comerciais é majoritariamente a identificação de PHP malware com bases em assinaturas. Dada as limitações dos antivírus comerciais, o estado-da-arte propõe extrair e analisar as características dos aplicativos malware através de ciência dos dados, máquinas de aprendizados estatísticos e Inteligência Artificial. A intenção do estado-da-arte é detectar o *cyber*-ataque antes mesmo dele alcançar o computador pessoal do cliente.

PEKTAS, *et al* (2017) emprega 17.900 arquivos maliciosos (Exe 32 bits, HTML, FLASH, Java e APK) do banco de dados VirusShare (VIRUSSHARE, 2019). O trabalho assegura o rótulo das amostras através da plataforma VirusTotal e usa uma técnica dinâmica para analisá-los, o que é feito através de duas ferramentas *sandboxes*: VirMon e Cuckoo. Inicialmente, os arquivos são enviados para a plataforma VirusTotal, para garantir que as amostras maliciosas estejam infectadas com códigos maliciosos. Depois disso as amostras são submetidas às ferramentas Cuckoo e VirMon, aos quais executarão os arquivos e criarão um relatório das ações executadas. Estes relatórios são analisados por diferentes algoritmos de aprendizado online (PA-I, PA-II, CW, AROW e NHERD) através da plataforma online jubatus. Nos algoritmos de aprendizagem *on-line* 5 pesos diferentes (1.0, 2.0, 3.0, 4.0 e 5.0) são utilizados, com validação cruzada de 10 k-*fold*. A sugestão é que a ferramenta Weka foi utilizada embora não haja tal relato no artigo. A obra de PEKTAS, *et al* (2017) é comparado com diferentes trabalhos contendo diferentes técnicas de classificação de arquivos maliciosos, ao qual obteve o quinto melhor resultado. Cabe ressaltar que é possível verificar que as acurácias, em média, obtiveram melhores resultados, o que indica ser uma metodologia promissora. Na obra de PEKTAS, *et al* (2017), a sensibilidade dos resultados é descrita através de uma matriz de confusão e do algoritmo CW. No melhor cenário, o trabalho de PEKTAS, *et al* (2017), atinge uma precisão de 94% no treinamento e 92,5% no teste (PEKTAS, *et al.*, 2017).

SESHAGIRI, *et al* (2016) emprega 789 Javascripts malignos oriundos da Malware Domain List e 1000 benignos da base de dados Alexa 500 (SESHAGIRI, *et al.*, 2016). O trabalho utiliza o Google Safe Browsing para validar a rotulagem das amostras benignas e malignas. É utilizado uma abordagem estática neste trabalho. Na análise estática é utilizado um algoritmo de árvore de decisão visando estimar a probabilidade de cada nó e classificá-los entre benigno e malware. As amostras são encaminhadas para uma árvore de decisão que irá verificar se nelas exista algum conteúdo classificado como maligna (e.g. lista negra). Caso possuam, a árvore de decisão irá calcular as probabilidades e caso esteja com um valor abaixo de 20% (definido de forma arbitrária), as amostras são rotuladas como verdadeiramente maliciosas. No estágio de classificação, não são descritas quaisquer informações de configuração dos classificadores e variações de parâmetros, portanto, supõe-se que os parâmetros empregados estejam relacionados às configurações padrões da ferramenta Weka. As bases de dados não são disponibilizadas o que também dificulta a reprodutibilidade dos testes. Neste artigo, não é demonstrada a sensibilidade do teste através de uma matriz de confusão ou curva ROC ao qual dificulta verificar o real desempenho do algoritmo na tarefa de classificação. O trabalho de SESHAGIRI, *et al* (2016) atinge uma acurácia de 89,44% (SESHAGIRI, *et al.*, 2016).

JAYASINGHE, *et al* (2014) emprega 10.620 javascripts malignos obtidos de diferentes sites e 10.620 benignos da base de dados Alexa 500 (JAYASINGHE, *et al.*, 2014). O trabalho utiliza o Google Safe Browsing para validar a rotulagem das amostras benignas e malignas. O trabalho utiliza uma abordagem dinâmica. Na análise dinâmica são utilizadas árvores de decisão, Classificador de Bayes e SVM (Máquinas de Vetor de Suporte) para classificar as amostras entre benignas e malware. Primeiramente, são retirados das páginas os códigos gerados durante a abertura da página. Esses códigos são convertidos para um formato inteligível para as máquinas de aprendizagem através de redução de dados, extração de características (feitas dinamicamente através da ferramenta ADSandbox) e representação de características. Em seguida, um classificador é treinado com as amostras obtidas e por fim novas páginas são apresentadas para o classificador a fim de verificar a sua assertividade. No estágio de classificação, a validação cruzada é implementada através do método k-fold, $k = 10$. A base de dados é dividida randomicamente em 10 partes iguais, contendo a mesma quantidade de amostras malignas e benignas. O critério utilizado para dividir a base em 10 partes não é descrito, o que nos leva a concluir que foi definido de forma arbitrária. São utilizados 3 diferentes classificadores (árvores de decisão, SVM e classificador de bayes). Neste artigo, a sensibilidade do teste é aferida através de uma matriz de confusão e gráficos boxplots. Logo, é possível verificar o real desempenho do algoritmo na tarefa de classificação. O trabalho de JAYASINGHE, *et al* (2014) atinge uma acurácia máxima de 96,55% na tarefa de classificação com a SVM (JAYASINGHE, *et al.*, 2014).

KAPLAN, *et al* (2013) emprega 563 Javascripts malignos e 3.954 benignos obtidos de diferentes fontes (KAPLAN, *et al.*, 2013). O trabalho não descreve nenhum método para validar o rótulo das amostras. É utilizado uma abordagem estática neste trabalho. Na análise estática é utilizado um classificador Bayesiano para classificar as amostras em benignas e malignas. As amostras são encaminhadas para o classificador que irá verificar se nelas existem alguma *string* classificado como maligna (e.g. lista negra). Caso possuam, o classificador irá rotular como maligna. No estágio de classificação, não são descritas quaisquer informações de configuração dos classificadores e variações de parâmetros, portanto, supõe-se que os parâmetros empregados estejam relacionados às configurações padrões da ferramenta Weka. Os resultados não são comparados com outras abordagens, o que dificulta medir a real efetividade da abordagem propostas. As bases de dados não são disponibilizadas o que também dificulta a reprodutibilidade dos testes. Neste artigo, não é demonstrada a sensibilidade do teste através de uma matriz de confusão ou curva ROC ao qual dificulta verificar o real desempenho do algoritmo na tarefa de classificação. O trabalho de KAPLAN, et al (2013) atinge uma acurácia de 99,00% na detecção de JavaScripts obfuscados (KAPLAN, *et al.*, 2013).

Quanto às bases de dados de arquivos malware, os trabalhos do estado da arte apenas informam as fontes de aquisição dos arquivos, no entanto, não há a descrição de quais arquivos são empregados nos experimentos. Logo, torna-se inviável a réplica das obras de PEKTAS, *et al* (2017), SESHAGIRI, *et al* (2016), JAYASINGHE, *et al* (2014) e KAPLAN, *et al* (2013). O trabalho proposto cria a base de dados PAEMAL cujo objetivo é dar total possibilidade da metodologia proposta ser replicada, por terceiros, em trabalhos futuros (PAEMAL, 2019). Logo, o artigo proposto, ao disponibilizar, livremente, a sua base de dados viabiliza transparência e imparcialidade à pesquisa, além de demonstrar a veracidade dos resultados alcançados.

Na etapa de extração de características, alguns trabalhos do estado-da-arte necessitam da presença local do arquivo visando a análise do seu código fonte. Enfatiza-se que *cyber*-ataque pode ser executado em um web servidor remoto ao invés do computador pessoal. Conclui-se que a extração de características estáticas é inválida mediante aplicativos malware executados no servidor visto que não há como periciar códigos fontes remotos sem a permissão do administrador do servidor web requisitado. Então, ao invés da inexequível análise estática, o nosso antivírus realiza a análise de comportamentos (dinâmica) do computador pessoal a partir do momento em que a página web foi requisitada.

Ao todo, nossa extração dinâmica de características monitora 11,777 comportamentos suspeitos no computador pessoal provocado pelo script PHP executado no servidor remoto. Inclui-se a perícia quanto à corrupção do Sistema Operacional e do navegador, além de perícia no tráfego de rede. Nossa solução NGAV é capaz de reconstruir uma cadeia de eventos, destrinchando a real intenção do *cyber*-ataque. Logo, o nosso antivírus não se atém a eventos individuais e discretos.

Na etapa de classificação entre arquivos benignos e malware, uma boa capacidade de generalização de técnicas de máquinas de aprendizado estatístico pode depender de uma boa escolha dos seus parâmetros de configuração. Em máquinas de aprendizado, não há um conjunto de parâmetros que satisfaça todos os tipos de aplicações (HUANG, *et al.*, 2012). A melhor combinação depende do conjunto de dados empregados (HUANG, *et al.*, 2012). Um método para identificar bons parâmetros consiste em treinar diferentes combinações das máquinas de aprendizado (HUANG, *et al.*, 2012). Então, o trabalho proposto explora diferentes parâmetros das máquinas de aprendizado. A hipótese é verificar se os classificadores sofrem alterações, em suas acurácias, em função das condições iniciais. Por outro lado, as obras de PEKTAS, *et al* (2017), SESHAGIRI, *et al* (2016), JAYASINGHE, *et al* (2014) e KAPLAN, *et al* (2013) não investigam diferentes parâmetros das máquinas de aprendizado, os autores apenas empregam as configurações padrões disponibilizadas pela ferramenta Weka empregada na etapa de classificação. Conclui-se que não há garantias das máquinas de aprendizado, empregadas pelo estado-da-arte, apresentarem resultados aceitáveis caso as configurações iniciais sejam desfavoráveis.

4. Materiais e Métodos

O presente trabalho visa elaborar a PAEMAL (***P**HP **A**nalysis **E**nvironment Applied to **Ma**lware Machine **L**earning* - Ambiente de Análise PHP Aplicado à Aprendizagem de Máquinas de Malware). A PAEMAL é uma base de dados a qual permite a classificação de arquivos PHP entre maliciosos e benignos. A PAEMAL é composta de 200 arquivos PHP malware e outros 1000 arquivos PHP benignos. Em relação às pragas virtuais, a PAEMAL extraiu arquivos PHP maliciosos do VirusShare o qual é um repositório de amostras de malware que proporciona acesso ao código malicioso real (VIRUSSHARE, 2019). Visando o catálogo dos 200 exemplares de PHP malware, foram necessárias a aquisição e análise, por scripts autorais, de cerca de 1.300.000 (1 milhão e trezentos mil) aplicativos malware a partir dos relatórios atualizados pelo VirusShare diariamente.

No que tange aos arquivos PHP benignos, o catálogo foi dado a partir dos scripts nativos de ferramentas *open source* a exemplo do *phpMyAdmin*. Enfatiza-se que todos os arquivos benignos foram submetidos à auditoria do VirusTotal. Logo, os 1000

exemplares de arquivos PHP benignos tiveram sua benevolência atestada pelos principais antivírus comerciais mundiais. Os resultados obtidos correspondentes às análises dos PHP benignos e malware, resultante da auditoria do VirusTotal, estão disponibilizados para consulta no endereço virtual do nosso repositório (PAEMAL, 2019).

Caso não houvesse qualquer tipo de tratamento na PAEMAL, haveria uma tendência de acertos maiores na classe majoritária (benigna) e elevada taxa de erro na classe minoritária (malware). A explicação é que, na base de dados PAEMAL, a quantidade de amostras benignas e malware são desiguais; 200 e 1000, respectivamente. Logo, ao empregar bases de dados desbalanceadas, as taxas de acerto dos classificadores podem ser favorecidas basta que eles sejam tendenciosos em relação à classe majoritária (AMOR, *et al.*, 2004). Visando não favorecer classificadores tendenciosos, o trabalho proposto emprega uma estratégia inspirada em trabalhos de engenharia biomédica. Na área de saúde, a presença de uma anormalidade (e.g. câncer) ocorre a cada milhares de diagnósticos de pacientes sadios. Então, a estratégia biomédica diz respeito a repetir o treinamento de acordo com a razão entre a classe majoritária e minoritária (200:1000 = 5 iterações) (WANG, *et al.*, 2017). No nosso trabalho, a cada iteração, um novo pacote da classe majoritária é apresentado à classe minoritária (200:200). Dessa forma, garante-se o não favorecimento a classificadores tendenciosos aliado à manutenção da diversidade das distintas amostras, da classe majoritária, contidas na base de dados (WANG, *et al.*, 2017).

Na prática clínica biomédica, a absorção de uma amostra maligna (e.g.: câncer) acarreta em um falso negativo. Vale salientar que as chances de recuperação da paciente estão associadas à detecção da doença de maneira precoce. Então, o trabalho proposto se inspira nos cuidados metodológicos tomados pelo estado-da-arte da engenharia biomédica no sentido de reservar quantidades relevantes de exemplares benignos e malware nas amostras separadas para o treinamento e teste. Logo, supondo uma amostra reservada à teste com pouca ou nenhuma instância da classe malware, logo a classificação, tendenciosa à classe benigna, teria sua taxa de acerto favorecida. Portanto, o trabalho proposto apresenta o cuidado metodológico de selecionar equitativamente, de forma randômica, exemplares benignos e malware para as amostras destinadas ao treinamento e teste.

O objetivo da criação da base de dados PAEMAL é dar total possibilidade da metodologia proposta ser replicada, por terceiros, em trabalhos futuros. Logo, o PAEMAL disponibiliza, livremente, de todas as suas amostras tanto benignas quanto malware:

- auditorias do VirusTotal,
- análises dinâmicas da nossa *Next Generation Sandbox*,

A PAEMAL também disponibiliza, em seu endereço virtual, seus 1000 arquivos PHP benignos. Além disso, a nossa base exibe a relação de todos os outros 200 arquivos PHP, dessa vez, malware. Então, há a possibilidade da aquisição de todos os aplicativos malware, empregados pela PAEMAL, através do estabelecimento de acordo e submissão às normas de uso do *VirusShare* (VIRUSSHARE, 2019). Conclui-se que a nossa base de dados PAEMAL viabiliza transparência e imparcialidade à pesquisa, além de demonstrar a veracidade dos resultados alcançados. Então, espera-se que o PAEMAL sirva de base para a criação de novos trabalhos científicos visando novos *Next Generation Antivirus*.

5. Metodologia

A Figura 3 exibe o diagrama da metodologia proposta em diagrama de blocos. Inicialmente, é criada uma aplicação web empregando um script PHP suspeito no servidor. Então, o cliente requisita a página Web suspeita do servidor. A partir daí, os comportamentos maliciosos, oriundos do ataque "sem arquivos", são auditados por nossa *Web-Server Next Generation Sandbox*. Na etapa seguinte, as características dinâmicas dos arquivos PHP são armazenadas num formato compatível com o aprendizado de máquina. Como método de extração de características, alguns comportamentos, auditados pela *Sandbox* são desprezados. O critério adotado de mineração diz respeito à eliminação de características as quais dizem respeito a um único arquivo PHP, como por exemplo, nomes de processo, hashes md5 e sha, dentre outros.

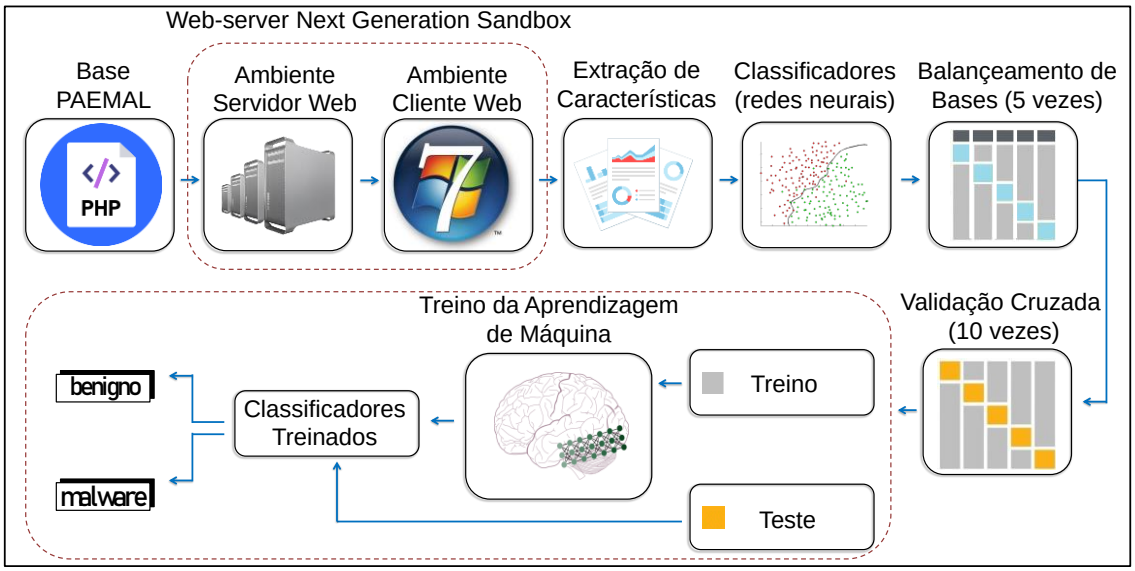

Figura 3. Diagrama da Metodologia Proposta.

A PAEMAL apresenta 200 e 1000 arquivos PHP malignos e benignos, respectivamente. Então, por cinco iterações, um pacote distinto de 200 exemplares da classe majoritária (benigna) é apresentado aos 200 exemplares da classe minoritária (malware). Após o balanceamento da base de dados, os comportamentos suspeitos dos arquivos PHP servem como atributos de entrada das redes neurais artificiais empregadas como classificadores. O objetivo é agrupar os arquivos PHP em duas classes; benignos e malware. Em cada combinação (200 benignos: 200 malware) oriundas do balanceamento da base de dados, é empregado o método de validação cruzada *k-fold*, onde k=10. A acurácia do classificador é a média aritmética da taxa de acertos obtida nas dez iterações.

De modo a validar o nosso NGAV, o trabalho proposto desenvolve um ambiente controlado nomeado *Web-Server Next Generation Sandbox*. Em nosso ambiente, são desenvolvidos o lado servidor e o cliente visando virtualizar o servidor web malicioso e o computador pessoal, respectivamente. O nosso servidor virtualizado é composto pelo Sistema operacional Linux, Servidor HTTP Apache, Servidor MySQL e interpretador PHP. No cliente, é empregado o Sistema Operacional Windows 7 dotado das instalações do Python, Máquina Virtual Java (JVM), Adobe Reader, Microsoft Office, e navegador Chrome. Em nossa *Web-Server Next Generation Sandbox*, os arquivos PHP, da nossa base de dados, são executados no servidor de modo a infectar, propositalmente, o cliente.

Então, os comportamentos maliciosos, oriundos do ataque “sem arquivos”, servem como atributos de entrada das máquinas de aprendizado estatístico.

5.1 Extração Dinâmica de Características

Em nossa metodologia, o malware é executado visando infectar, propositalmente, o Windows 7 auditado, em tempo real (dinâmico). Na nossa *Web-Server Next Generation Sandbox,* a quantidade de características está em função do comportamento dinâmico do sistema. Em média, são geradas 11.777 características referentes ao monitoramento do arquivo suspeito no ambiente controlado proposto. A seguir, são detalhados os grupos de características referentes ao monitoramento do sistema.

- ✓ Características relacionadas à Injeção de código, técnica usada por um invasor para introduzir código em programas vulneráveis e mudar seu comportamento. A auditória verifica se o servidor requisitado tenta:
  - executar um processo e injetar código enquanto é descompactado;
  - injetar código em um processo remoto com o uso de uma das seguintes funções: *CreateRemoteThread* ou *NtQueueApcThread*.
- ✓ Características relacionadas a Keyloggers, programas que gravam todas as entradas de teclado feitas pelo usuário, com a finalidade principal de capturar de forma ilegal senhas e outras informações confidenciais. Verifica se o servidor investigado tenta:
  - criar mutexes dos keyloggers *Ardamax* ou *Jintor*.
- ✓ Características relacionadas à busca de outros programas possivelmente instalados. O objetivo é verificar se o servidor auditado busca:
  - descobrir onde o browser está instalado, caso exista algum no sistema;
  - descobrir se existe algum *sniffer* ou algum analisador de pacotes de rede instalado.
- ✓ Características relacionadas a desabilitar os componentes do Windows:
  - Verifica se o servidor testado tenta desabilitar algum dos programas do Windows: terminal de comandos, gerenciador de dispositivos ou Registro (Regedit).
- ✓ Características relacionadas à forense de memória, processo em que o conteúdo da memória RAM é periciado para fins de diagnóstico. A forense digital proposta audita se o servidor requisitado tenta:
  - encontrar URLs maliciosas no processamento da forense da memória;
  - encontrar evidências da presença e uso do programa Yara. Tal programa é, em regra geral, empregado para realizar a forense digital de memória.
- ✓ Características relacionadas a mineração de cripto-moedas:
  - O nosso antivírus audita se o servidor invocado tenta se conectar a *pools* de mineração, com a finalidade de gerar moedas virtuais sem o conhecimento (e sem beneficiar) o proprietário do computador.
- ✓ Características relacionadas a modificações no sistema:

- O antivírus proposto verifica se o servidor demandado tenta criar ou modificar certificados de sistema, avisos do centro de segurança, comportamentos de controle de contas de usuário, papel de parede da área de trabalho ou valores do *ZoneTransfer.ZoneID* no identificador de zona ADS (*Alternate Data Stream*).

✓ Características relacionadas ao *Microsoft Office*. O nosso antivírus verifica se o servidor requisitado tenta:

- criar um objeto atrelado à linguagem de programação *Visual Basic*;
- executar processos do *Microsoft Office* inseridos em um objeto de interface de linha de comando empacotado.

✓ Características relacionadas a empacotamento e obfuscação. A forense digital proposta verifica se o servidor demandado:

- possui informação compactada ou criptografada indicativa de empacotamento;
- cria uma cópia ligeiramente modificada dele mesmo (pacote polimórfico);
- é compactado utilizando UPX (*Ultimate Packer for Executables*) ou VMProtect (software utilizado para obfuscar código e virtualizar programas).

- Características relacionadas à persistência. Cabe ressaltar que a vítima pode não estar livre da infecção de um malware mesmo após a sua detecção e eliminação. A persistência das malfeitorias pode ocorrer mesmo após a exclusão do malware (LIMA, *et al.*, 2018). Logo, quando o sistema operacional é inicializado, o *cyber*-ataque recomeça devido ao recomeço da vulnerabilidade explorada pelo malware (e.g.: redirecionar a página inicial do Internet Explorer). Logo, o antivírus criado audita se o servidor requisitado tenta:

  - usar Javascript em um valor de chave de registro no Registro do Sistema (Regedit);
  - instalar um auto-executável na iniciação do Windows (*autorun*);
  - instalar um executável nativo para ser executado na inicialização do Windows.

✓ Características relacionadas a POS (*Point of Sale*), tipo de ataque que visa obter as informações de cartões de crédito e de débito das vítimas. O antivírus criado audita se o servidor invocado tenta:

- criar arquivos relacionados ao malware POS Alina;
- contactar servidores relacionados ao malware POS Alina;
- contactar botnets relacionadas ao malware POS blackpos;
- criar mutexes relacionados ao malware POS decebel;
- criar mutexes e chaves de registro relacionados ao malware POS Dexter;
- criar mutexes e chaves de registro relacionados ao malware POS jackpos;
- contactar botnets relacionadas ao malware POS jackpos;
- contactar servidores relacionados ao malware POS poscardstealer;

✓ Características relacionadas à injetores de códigos de *powershell*. O nosso antivírus verifica se o servidor requisitado:

- é um script de powershell malware do tipo powerfun ou powerworm.
- tenta criar um processo powershell suspeito;
- tenta criar entradas de registros via scripts powershell;

- ✓ Características relacionadas aos processos. O antivírus proposto verifica se o servidor invocado:
  - interessa-se em algum processo específico em execução;
  - procura repetidas vezes por um processo não encontrado;
  - tenta falhar algum processo em específico.
- ✓ Características relacionadas a *ransomwares*, ataques que tornam os dados do equipamento inacessíveis, exigindo pagamento para restabelecer o acesso do usuário. O nosso antivírus verifica se o servidor requisitado tenta:
  - criar mutexes do *ransomware* chanitor;
  - executar comandos no bcdedit (ferramenta de linha de comando que gerencia dados de configuração de inicialização) relacionados à ransomware;
  - adicionar extensões de arquivos reconhecidamente relacionadas à *ransomwares* a arquivos que foram criptografados;
  - executar movimentações em arquivos, que podem ser indicativos do processo de encriptação de dados visto em um ataque *ransomware*;
  - criar instruções de como reverter a criptografia feita em um ataque *ransomware* ou se tenta gerar um arquivo de chave;
  - escrever uma mensagem de resgate em disco, provavelmente associada a um ataque *ransomware*;
  - esvaziar a lixeira;
  - remover ou desabilitar o *shadow copy*, recurso que tem como finalidade agilizar a restauração de dados, a fim de evitar a recuperação do sistema.
- ✓ Características relacionadas ao uso de sandboxes. A forense digital averigua se o servidor invocado tenta:
  - detectar se as sandboxes: Cuckoo, Joe, Anubis, Sunbelt, ThreatTrack/GFI/CW ou Fortinet estão sendo utilizadas, por meio da presença de arquivos próprios utilizados por elas;
  - procurar por diretórios conhecidos onde uma sandbox pode executar amostras;
  - checar se alguma atividade humana está sendo desempenhada;
  - descobrir o tempo em espera do Windows, a fim de determinar o tempo total de atividade do Windows;
  - instalar um procedimento que monitora eventos do mouse;
  - desligar ou reiniciar o sistema visando burlar a *sandbox*;
  - atrasar as tarefas de análise;
  - desligar funções do Windows monitoradas pela *Cuckoo sandbox*.
- ✓ Características relacionadas ao Registro (Regedit) do Windows 7 SO:
  - Mudanças nas associações entre extensões de arquivos e conjunto de *software* instalado na máquina (HKEY_CLASSES_ROOT).
  - Modificações nas informações sobre o usuário atual (HKEY_CURRENT_USER).
  - Corrupção do funcionamento dos drivers (HKEY_LOCAL_MACHINE).
  - Alterações nas configurações de aparência do Windows e as configurações efetuadas pelos usuários, como papel de parede, protetor de tela e temas (HKEY_USERS).

- Mudanças nas Configurações de hardware (HKEY_CURRENT_CONFIG).

✓ Características relacionadas a cavalos de troia (programa malicioso que entra em um computador disfarçado como outro programa, legítimo) de acesso remoto, ou RAT (Remote Access Trojans). O nosso antivírus verifica se o servidor requisitado tenta criar arquivos, chaves de registro e/ou mutexes relacionados aos RATs: Adzok, bandook, beastdoor, beebus, bifrose, blackhole/schwarzesonne, blackice, blackshades, bladabindi, bottilda, bozokrat, buzus, comrat, cybergate, darkcloud, darkshell, delf trojan, dibik/shark, evilbot, farfli, fexel, flystudio, fynloski/darkcomet, ghostbot, hesperbot, hkit backdoor, hupigon, icepoint, jewdo backdoor, jorik trojan, karakum/saharabot, koutodoor, aspxor/kuluoz, likseput, madness, madness, magania, minerbot, mybot, naid backdoor, nakbot, netobserve spyware, netshadow, netwire, nitol/servstart, njrat, pasta trojan, pcclient, plugx, poebot/zorenium, poison ivy, pincav/qakbot, rbot, renos trojan, sadbot, senna spy, shadowbot, siggen, spynet, spyrecorder, staser, swrort, travnet, tr0gbot bifrose, turkojan, urlspy, urx botnet, vertexnet, wakbot, xtreme, zegost.

✓ Características relacionadas ao *payload* na rede. O nosso antivírus audita se o servidor invocado tenta:

- verificar se a atividade de rede contém mais de um *useragent* único;
- criar mutexes de protocolo de conexão de área de trabalho remota (RDP);
- checar a presença de clientes de chat mIRC;
- instalar Tor (the onion router, software de código aberto com a capacidade de criar de forma segura e anonimamente conexões online, a fim de resguardar o direito à privacidade do usuário), ou um serviço oculto Tor na máquina;
- conectar a um encurtador de URL chinês com histórico malicioso;
- criar mutexes relacionados a ferramentas de administração remota VNC (Virtual Remote Computer).

✓ Características relacionadas ao tráfego de rede. Audita-se se o servidor suspeito tenta:

- conectar-se a um IP que não está mais respondendo a requisições;
- resolver um domínio de topo suspeito;
- iniciar a escuta (socket) com algum servidor;
- conectar a algum domínio de DNS dinâmico;
- fazer requisições de HTTP;
- gerar tráfego ICMP;
- conectar-se a algum servidor de IRC (possivelmente parte de alguma botnet);
- fazer requisições SMTP (possivelmente envio de SPAM);
- conectar-se a algum serviço oculto TOR por meio de um gateway TOR;
- iniciar o arquivo wscript.exe, que pode indicar um script baseado em download de *payload* (corpo do pacote);
- gerar alertas IDS ou IPS, com Snort e Suricata (ferramentas de gerenciamento e monitoramento de redes).

✓ Características relacionadas à servidores DNS (Domain Name System, servidores responsáveis pela tradução de endereços URL em IP). O nosso antivírus investiga se o servidor requisitado tenta:

- conectar a servidores DNS de provedores de DNS dinâmicos;
- conectar ao site malicioso expirado 3322.org, ou ao seu domínio relacionado, 125.77.199.30;
- resolver algum domínio *Free Hosting*, possivelmente malicioso.

5.2 Redes Neurais visando o Reconhecimento de Padrão de Malware

Redes neurais são modelos, de inteligência computacional, utilizadas para resolver problemas de classificação tendo como principal característica o poder de generalização diante de dados não apresentados à rede. A rede ELM (*Extreme Learning Machine* – Máquina de Aprendizado Extremo) tem como principal característica a velocidade de treinamento e predição de dados comparada a outros classificadores (HUANG, *et al.*, 2012). As ELMs têm sido largamente aplicadas nas mais diversas áreas como na Engenharia Biomédica (AZEVEDO, *et al.*, 2015) (AZEVEDO, *et al.*, 2015b) (LIMA, *et al.*, 2016) (LIMA, *et al.*, 2014) (CORDEIRO, *et al.*, 2012).

As redes ELMs podem contribuir bastante para o avanço da segurança em dispositivos visto que a inteligência artificial ainda se encontra em um estágio inicial na área de Segurança da Informação (HENKE, *et al*., 2011). A rede ELM tem como principal característica a velocidade de treinamento e predição de dados comparada a outros classificadores (HUANG, ZHOU, *et al.*, 2012). A rede ELM é uma rede de camada escondida única, não recursiva. O processo de aprendizagem da rede ELM é baseado na inversa generalizada de Moore-Penrose (pseudo-inversa), onde são calculados os pesos entre a camada escondida e a camada de saída (HUANG, ZHOU, *et al.*, 2012).

A aprendizagem da rede ELM é realizada em lote, onde todos os dados são apresentados à rede antes do ajuste dos pesos referentes às ligações sinápticas entre os neurônios da camada escondida e de saída. Há uma única iteração, tornando o treinamento mais rápido do que as abordagens convencionais. Então, não é necessário determinar o máximo número de iterações, uma vez que o algoritmo não é iterativo. Além disso, por não se basear no método de gradiente descendente, a rede não sofre o problema de mínimo local nem é necessária a definição de um parâmetro de taxa de aprendizagem.

Matematicamente, na rede ELM os atributos de entrada x_{ik} correspondem ao conjunto $\{x_{it} \in R;\ i \in N^*,\ i = 1, \dots, n;\ t \in N^*,\ t = 1, .. v\}$. Logo, há *n* características extraídas da aplicação e v vetores de dados de treinamento. A camada escondida h_j, constituída por m neurônios, é representada pelo conjunto $\{h_j \in R;\ j \in N^*,\ j = 1, \dots, m\}$.

O processo de treinamento da ELM é rápido por ser composto por poucas etapas. Inicialmente, os pesos de entrada w_{ji} e $biasb_{jt}$ são definidos de maneira aleatória. Dada uma função de ativação $f: \mathbb{R} \to \mathbb{R}$, o processo de aprendizagem é dividido em três passos:

1. Atribuição aleatória de pesos w_{ji}, correspondente aos pesos entre a camada de entrada e a camada escondida, e *bias* b_{jk}.
2. Calcular a matriz H, que corresponde à saída dos neurônios da camada escondida.

3. Calcular a matriz dos pesos de saída $\beta = H^{\dagger}Y$, onde $H^{\dagger}$ é a matriz inversa generalizada de Moore-Penrose da matriz H, e Y corresponde à matriz de saídas desejadas *s*.

A saída dos neurônios da camada escondida, correspondente à matriz H, é calculada através da função de ativação, entradas e pesos da camada escondida, conforme mostra a Equação (1).

$$H_{jt} = \begin{bmatrix} K(11) & \cdots & K(1N) \\ \vdots & \ddots & \vdots \\ K(V1) & \cdots & K(VN) \end{bmatrix} \quad (1)$$

Diferente das redes com retropropagação, na rede ELM não é necessário definir critério de parada para treinamento nem criar mecanismos para que a rede não perca a capacidade de generalização. O motivo é a rede ELM apresenta uma única iteração. Desse modo, não é necessária a separação de conjunto de dados em treinamento, validação e teste. Basta a divisão em conjuntos de treinamento e teste, permitindo um maior número de amostras para esses dois conjuntos em comparação a redes neurais baseadas em retropropagação. Uma vez treinada a rede, os padrões de teste são apresentados juntamente com a saída desejada. A rede não sofrerá mais ajustes e apenas calculará o resultado obtido para cada conjunto de teste apresentado. Ao comparar os dados esperados com os obtidos é avaliado o grau de precisão da rede ELM.

O trabalho explora 9 (nove) tipos distintos de *kernels* visando redes neurais ELMs. No estado-da-arte, HUANG, *et al*, (2012) descreve 7 (sete) desses *kernels*; Linear, Polinomial, Transformada *Wavelets*, Sigmoid, Senoidal, *Hard* Limite e Tribas (*Triangular Base Function*). Além disso, são empregados dois outros *kernels* validados no campo da Engenharia Biomédica: *Fuzzy*-Dilatação e *Fuzzy*-Erosão (AZEVEDO, *et al.*, 2015) (AZEVEDO, *et al.*, 2015b).

Os *kernels* Polinomial, *Wavelets* e Linear não empregam camadas escondidas (HUANG, *et al.*, 2012). Nesses *kernels*, os cálculos são baseados na transformação dos dados de entrada e podem trabalhar de maneira aproximada dos *kernels* contendo arquiteturas dotadas de camadas escondidas (HUANG, *et al.*, 2012). Nos referidos *kernels*, uma boa capacidade de generalização da rede ELM depende de uma escolha ajustada dos parâmetros (C, γ). Então, há a investigação dos parâmetros (C, γ) inspirada no método, proposto por HUANG, *et al*, (2012), que consiste em treinar sequencias crescentes de C e γ. Matematicamente, 2^n, onde $n = \{-24, 10, 0, 10, 25\}$. A hipótese é verificar se esses parâmetros com valores diferentes aos padrões; $(C = 1,\ \gamma = 1)$, geram resultados superiores. No *kernel* Linear, há a investigação apenas do parâmetro de custo C visto que não cabe a exploração do parâmetro do kernel γ (HUANG, *et al.*, 2012).

Os *kernels* Sigmoidal, Senoidal, *Hard* Limite, *Tribas*, *Fuzzy*-Dilatação e *Fuzzy*-Erosão empregam arquiteturas dotadas de camadas escondidas. Então, há a investigação quanto à quantidade de neurônios na camada escondida desses *kernels*. A hipótese é verificar se arquiteturas que exijam um maior volume de cálculos, como por exemplo, dobrar a quantidade de neurônios na camada escondida, são capazes de gerar taxas de acertos superiores em comparação a arquiteturas que exijam uma menor quantidade de cálculos. Há a avaliação de 2 (duas) arquiteturas, elas empregam 100 e 500 neurônios em suas respectivas camadas escondidas. Nós investigamos 30 conjuntos diferentes de pesos

iniciais referentes às ligações sinápticas entre os neurônios. A semente do gerador aleatório varia de 1 a 30 incrementalmente.

6. Resultados

A Tabela 3 detalha os resultados obtidos pelas redes neurais ELM através dos *kernels Wavelets* e Polinomial. Em cada *kernel*, por 5 vezes, um pacote distinto de exemplares benignos (classe majoritária) é apresentado ao pacote de exemplares malware (classe minoritária). Em cada uma dessas 5 vezes, há a validação cruzada através do método *k-fold* onde $k = 10$. Então, há 50 (5*10) iterações em cada linha da Tabela 3. Em relação à precisão na fase de teste, o melhor desempenho médio foi de 97,50% através do *kernel* Polinomial dotado dos parâmetros $(C, \gamma) = (2^{-10}, 2^{0})$. Nessa nossa melhor configuração, o tempo de treinamento médio é de apenas 0,04 segundos acompanhado de um desvio padrão de 0,05. Na Tabela 3, na Tabela 4 e na Tabela 5, há apenas a descrição dos melhores e piores casos, nessa ordem, para cada *kernel* ELM.

Tabela 3. Resultado das redes ELMs. Os parâmetros (C, γ) variam de acordo com o conjunto $\{2^{-24}, 2^{-10}, 2^{0}, 2^{10}, 2^{25}\}$.

Kernel	(C, γ)	*Acerto no treino (%)*	*Acerto no teste (%)*	*Tempo de treino (seg.)*	*Tempo de teste (seg.)*
Polynomial	$(2^{-10}, 2^{0})$	97,40 ± 2,30	**97,50 ± 4,32**	**0,04 ± 0,05**	**0,03 ± 0,03**
	$(2^{-24}, 2^{10})$	49,95 ± 0,11	49,95 ± 0,34	0,09 ± 0,10	0,04 ± 0,05
Wavelets	$(2^{10}, 2^{0})$	**100,00 ± 0,00**	66,59 ± 12,39	0,07 ± 0,08	0,05 ± 0,06
	$(2^{-24}, 2^{-24})$	**100,00 ± 0,00**	52,35 ± 4,35	0,07 ± 0,09	0,05 ± 0,06

Tabela 4. Resultado das redes ELMs dotadas do *kernel* Linear. O parâmetro C varia de acordo com o conjunto $\{2^{-24}, 2^{-10}, 2^{0}, 2^{10}, 2^{25}\}$.

Kernel	C	*Acerto no treino (%)*	*Acerto no teste (%)*	*Tempo de treino (seg.)*	*Tempo de teste (seg.)*
Linear	2^{-24}	**98.22 ± 1.24**	**97.30 ± 5.51**	**0.04 ± 0.05**	0.03 ± 0.04
	2^{10}	99.95 ± 0.11	87.30 ± 14.04	**0.04 ± 0.05**	**0.03 ± 0.03**

A Tabela 4 exibe os resultados alcançados pelas redes ELMs dotadas do *kernel* Linear. Há apenas a investigação do parâmetro de custo C, não é possível a exploração do parâmetro γ em um *kernel* Linear (HUANG, *et al.*, 2012). Cada linha na Tabela 4 contém 50 iterações assim como ocorreu na Tabela 3. Em relação à precisão na fase de teste, a máxima e mínima precisão média foi de 97,30% e 87,30%, respectivamente. Conclui-se que o parâmetro de custo C é capaz de aperfeiçoar o desempenho do *kernel* Linear quando aplicado à detecção de malware.

Tabela 5 detalha os resultados obtidos pelas redes neurais ELMs dotadas de camada escondida. Em cada kernel, por 5 vezes, um pacote separado de amostras benignas (classe majoritária) é apresentado ao pacote de amostras de malware (classe minoritária). Em cada uma dessas 5 vezes, investigamos 30 conjuntos diferentes de pesos iniciais referentes às ligações sinápticas entre os neurônios. A semente do gerador aleatório varia de 1 a 30 incrementalmente. Então, para cada conjunto de pesos sinápticos é empregado o método k-fold, onde $k = 10$. Então, há 1500 (5 * 30 * 10) iterações em cada linha da Tabela 5. Com relação à precisão, o desempenho médio máximo foi de 81,99% através do kernel de Fuzzy-Erosão dotado de 500 neurônios em sua camada escondida.

A Figura 4 e Figura 5 são representações gráficas dos resultados descritos na Tabela 3, na Tabela 4 e na Tabela 5. A Figura 4 (a) e a Figura 4 (b) mostram os *boxplots* para uma precisão ótima durante a fase de treinamento e teste, respectivamente. As taxas de acertos ótimas e suas configurações estão presentes nas primeiras linhas de cada kernel presentes na Tabela 3, na Tabela 4 e na Tabea 5.

Tabela 5. Resultados das redes ELMs. O número de neurônios da camada escondida varia de acordo com o conjunto {100, 500}.

Kernel	Neurons	*Acerto no treino (%)*	*Acerto no teste (%)*	*Tempo de treino (seg.)*	*Tempo de teste (seg.)*
Sigmoid	500	77.37 ± 22.25	61.05 ± 14.08	0.11 ± 0.11	0.01 ± 0.02
	100	77.18 ± 22.10	60.71 ± 14.83	0.03 ± 0.03	**0.00 ± 0.01**
Seno	500	**100.00 ± 0.00**	59.12 ± 14.41	0.12 ± 0.12	0.01 ± 0.02
	100	89.44 ± 13.02	58.48 ± 14.20	0.03 ± 0.03	**0.00 ± 0.01**
Hard limite	100	50.05 ± 0.11	50.05 ± 0.33	**0.02 ± 0.03**	**0.00 ± 0.01**
	500	50.05 ± 0.11	50.05 ± 0.33	0.12 ± 0.11	0.01 ± 0.02
Tribas	100	50.18 ± 0.41	49.77 ± 1.36	**0.02 ± 0.03**	**0.00 ± 0.01**
	500	50.46 ± 0.55	49.61 ± 2.14	0.10 ± 0.10	0.01 ± 0.02
Fuzzy-Dilatação	100	100.00 ± 0.02	80.64 ± 18.85	**0.02 ± 0.03**	**0.00 ± 0.01**
	500	**100.00 ± 0.00**	77.25 ± 15.57	0.12 ± 0.12	0.01 ± 0.02
Fuzzy-Erosão	500	**100.00 ± 0.00**	**81.99 ± 17.03**	0.15 ± 0.14	0.04 ± 0.04
	100	**100.00 ± 0.00**	80.75 ± 18.76	0.04 ± 0.04	0.01 ± 0.02

A Figura 4 (a) exibe as acurarias resultantes do treinamento. Entre as melhores configurações de cada *kernel*, a melhor e pior precisão média foram de 100,00% e 50,05% através dos *kernels Wavelets* e *Hard* Limite, respectivamente. A Figura 4 (b) exibe as acurarias resultantes da fase de teste. Entre as melhores configurações de cada kernel, a melhor e pior precisão média foram de 97,50% e 49,77% através dos kernels Polinomial e Tribas (*Triangular Basis Function*), respectivamente. Conclui-se que, na fase de teste, a melhor abordagem é quase 100% superior ao pior cenário possível. Portanto, a escolha de um kernel adequado, composto por uma arquitetura correta, é essencial para maximizar a acurácia quanto à identificação de malware.

A Figura 5 (a) e a Figura 5 (b) mostram os boxplots dos tempos gastos durante as fases de treinamento e teste, respectivamente. Em relação ao tempo de treinamento, o nosso kernel *Fuzzy*-Erosão, contendo 500 neurônios em sua camada escondida, é mais lento em relação aos demais, quando houve o consumo médio de 0,15 segundos. Por outro lado, os *kernels Hard* limite, Tribas e *Fuzzy*-Dilatação, contendo 100 neurônios em suas camadas escondidas, apresentam o treinamento mais rápido, quando houve um consumo médio de 0,02 segundos. Em relação ao tempo consumido, durante a fase de teste, não há grande discrepâncias entre os *kernels*.

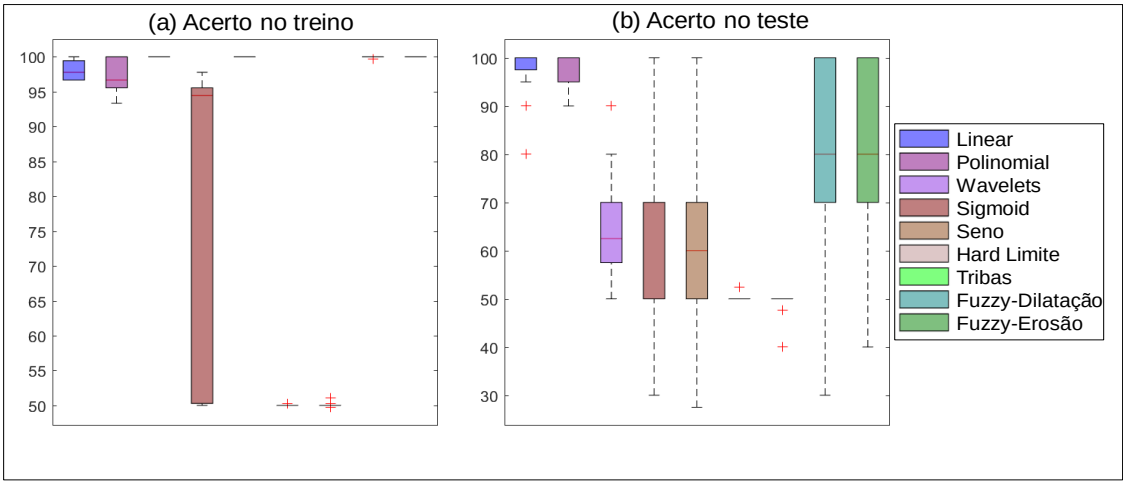

Figura 4. (a) Boxplot referente à acurácia de treinamento. (b) Boxplot referente à acurácia de teste.

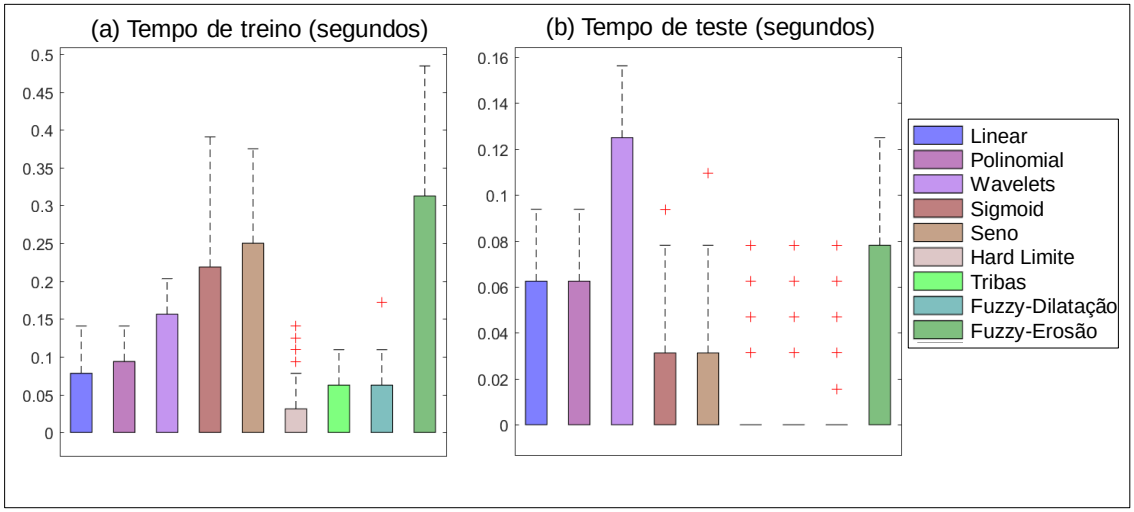

Figura 5. (a) Boxplot referente ao tempo gasto durante o treinamento. (b) Boxplot referente ao tempo consumido durante à fase de teste.

A Tabela 6 exibe as matrizes de confusão das melhores configurações das redes ELMs apresentadas na Tabela 3, na Tabela 4 e na Tabela 5. A matriz de confusão assume papel importante no sentido verificar a qualidade de uma aprendizagem supervisionada. Na

Tabela 6, “B” e “M” são abreviaturas de Benigno e Malware. As classes desejadas estão dispostas no rótulo vertical enquanto as classes obtidas estão no rótulo horizontal. Na

Tabela 6, por exemplo, o *kernel* Polinomial classificou em média, de maneira equivocada, 2,28 casos como benignos quando se tratavam de malware. Ainda quanto ao *kernel* Polinomial, houve a classificação média de 0,06 casos equivocadamente ditos como malware quando se tratavam de amostras benignas. Na matriz de confusão, a diagonal principal é ocupada por casos onde a classe obtida coincide com a classe desejada. Então, um bom classificador deve ter uma diagonal principal ocupada por valores altos enquanto as demais posições devem possuir valores baixos. Na Tabela 6, as diagonais principais estão em negrito.

Tabela 6 Matrizes de confusão das redes neurais ELMs apresentadas na Tabela 3, na Tabela 4 e na Tabela 5

.

. *Kernel*	*Melhor Configuração*		*Treino*		*Teste*	
			B	M	B	M
Polinomial	$(C, \gamma) = (2^{-10}, 2^{0})$	B	**98.94 ± 66.86**	0.06 ± 0.24	**11.00 ± 7.42**	0.00 ± 0.00
		M	2.28 ± 2.06	**97.08 ± 69.14**	0.28 ± 0.50	**10.76 ± 7.64**
Wavelets	$C = (2^{10}, 2^{0})$	B	**99.00 ± 66.81**	0.00 ± 0.00	**10.64 ± 7.75**	0.36 ± 0.69
		M	0.00 ± 0.00	**99.36 ± 67.26**	8.16 ± 7.58	**2.88 ± 1.12**
Linear	$C = 2^{-24}$	B	**98.88 ± 66.91**	0.12 ± 0.48	**10.94 ± 7.48**	0.06 ± 0.24
		M	2.16 ± 0.98	**97.20 ± 67.19**	0.30 ± 0.61	**10.74 ± 7.50**
Sigmoid	$neurônios = 500$	B	**28.18 ± 29.36**	70.82 ± 85.51	**2.34 ± 3.10**	8.66 ± 8.93
		M	3.58 ± 25.07	**95.78 ± 66.98**	1.27 ± 3.12	**9.77 ± 8.06**
Seno	$neurônios = 500$	B	**99.00 ± 66.16**	0.00 ± 0.00	**6.21 ± 3.46**	4.79 ± 4.56
		M	0.00 ± 0.00	**99.36 ± 66.60**	5.30 ± 4.19	**5.74 ± 3.99**
Hard Limite	$neurônios = 100$	B	**0.00 ± 0.00**	99.00 ± 66.16	**0.00 ± 0.00**	11.00 ± 7.35
		M	0.00 ± 0.00	**99.36 ± 66.60**	0.00 ± 0.00	**11.04 ± 7.41**
Tribas	$neurônios = 100$	B	**99.00 ± 66.16**	0.00 ± 0.00	**10.98 ± 7.37**	0.02 ± 0.13
		M	99.02 ± 66.51	**0.34 ± 0.47**	11.04 ± 7.41	**0.00 ± 0.00**
Fuzzy-Dilatação	$neurônios = 100$	B	**99.00 ± 66.16**	0.00 ± 0.00	**10.26 ± 7.98**	0.74 ± 0.90
		M	0.01 ± 0.08	**99.35 ± 66.59**	1.22 ± 1.22	**9.82 ± 8.39**
Fuzzy-Erosão	$neurônios = 500$	B	**99.00 ± 66.16**	0.00 ± 0.00	**10.35 ± 7.90**	0.65 ± 0.79
		M	0.00 ± 0.00	**99.36 ± 66.60**	1.19 ± 1.09	**9.85 ± 8.31**

A Tabela 7 disponibiliza os testes de hipóteses *t-students* (paramétrico) e Wilcoxon (não-paramétrico). Os testes se dão entre o *kern*el Polinomial e todos os demais investigados. As amostras dizem respeito à precisão durante a fase de teste com os kernels configurados em seus parâmetros ótimos apresentados nas primeiras linhas de cada *kernel*, na Tabela 3, na Tabela 4 e na Tabela 5. Observa-se que a escolha do *kernel* Polinomial foi devido ao fato dele ter alcançado a melhor precisão média, durante a fase de teste, dentre os *kernels* ELMs. O p-valor do teste é um valor escalar no intervalo [0,1]. p é a probabilidade do teste estatístico apresentar o valor observado ou algo mais extremo. Se p apresentar um valor baixo, o resultado observado é estatisticamente relevante. Caso o teste de hipótese seja 1, a hipótese nula é rejeitada, portanto, as distribuições são diferentes.

Ao observar a Tabela 7, os resultados do *kernel* Polinomial são confrontados contra todas as demais amostras. A hipótese foi rejeitada em quase todos os casos. Logo, em regra geral, o *kernel* Polinomial é estatisticamente distinto aos demais. A explicação é que a hipótese foi igual a 1 tanto no teste paramétrico quanto no não-paramétrico. A exceção ocorreu envolvendo os *kernels* Polinomial e Linear. Nessa exceção, a hipótese nula foi aceita. Então, os *kernels* Polinomial e Linear geram resultados estatisticamente equivalentes tanto no teste paramétrico (*t-students*) quanto no teste não-paramétrico (*Wilcoxon*).

Tabela 7: Teste de hipóteses *t*-students e Wilcoxon entre a melhor configuração (*kernel* Polinomial) e todas as demais.

Comparação	*t-students (teste paramétrico)*		*Wilcoxon (teste não-paramétrico)*	
	Hipótese	***Valor p***	***Hipótese***	***Valor p***
Polinomial *vs Wavelets*	1	2.08114e-22	1	6.46726e-18
Polinomial *vs* Linear	0	0.778313	0	0.894246
Polinomial *vs Sigmoid*	1	0.000000	1	0.000000
Polinomial *vs* Seno	1	0.000000	1	0.000000
Polinomial *vs Hard* limite	1	0.000000	1	0.000000
Polinomial *vs Fuzy*-Dilatação	1	4.91637e-194	1	6.91101e-148
Polinomial *vs Fuzzy*-Erosão	1	6.2273e-196	1	2.1272e-141

7. Conclusão

Apesar da presença, quase totalitária, dos antivírus nos computadores pessoais, os aplicativos malware vêm causando prejuízos bilionários e em escalas cada vez maiores (MICROSOFT, 2017). Uma das explicações é que os *cyber*-ataques se renovam sistematicamente (SOPHOS, 2014). Visando suprir as limitações dos antivírus comerciais, o estado-da-arte emprega a análise do código-fonte do arquivo suspeito, conhecida como análise estática, de modo que o repertório de instruções pode ser estudado. Portanto, é possível investigar a intenção maliciosa do arquivo antes mesmo dele ser executado pelo usuário (LIMA, *et al.*, 2018). A análise estática, no entanto, é impraticável mediante ataques "sem arquivos" visto que o arquivo é executado remotamente e, portanto, o executável não está presente no computador pessoal.

Ao invés da inexequível análise estática, a extração de características do nosso NGAV diz respeito à perícia do comportamento anômalo, no computador da vítima. Em média, nossa extração dinâmica de características monitora 11,777 comportamentos que o ataque "sem arquivos" possa fazer quando diretamente lançado de um servidor malicioso para um serviço responsivo em um computador pessoal. Ao invés de analisar eventos individuais, nossa solução consegue reconstruir a cadeia de eventos tal qual o malfeitor lançaria.

Nesse trabalho, são aplicadas máquinas de aprendizado do tipo ELM na perícia forense digital especificamente no reconhecimento de padrão de malware. Então, os comportamentos maliciosos, obtidos através da nossa *Web-server Next Generation Sandbox*, servem como atributos de entrada das máquinas de aprendizado estatístico empregadas como classificadores. A meta é agrupar os arquivos em duas classes: benigna e malware. Quanto à precisão, o *kernel* Polinomial demonstrou melhor desempenho em relação aos demais classificadores analisados, e obteve um acerto médio de 97,50%. Essa abordagem possui os seguintes parâmetros $(C, \gamma) = (2^{-10}, 2^{0})$. Por outro lado, o kernel Tribas (*Triangular Basis Function*) obteve a pior acurácia com desempenho médio de 49,77% mesmo em sua melhor configuração. Esse caso com menor precisão se dá quando o *kernel* Tribas faz uso de 100 neurônios na sua camada escondida. Conclui-se que a melhor abordagem é superior em quase 100% em comparação ao pior cenário. Logo, a

escolha de uma adequada função de aprendizado, composta por corretos parâmetros, é essencial para maximizar a precisão quanto à identificação de malware.

O NGAV proposto pode ser estendido no sentido de prover *cyber*-proteção a redes locais. Logo, a meta futura é que o nosso NGAV possa ser executado tanto nos computadores pessoais quanto no servidor *proxy* o qual se constitui como o intermediário entre a rede mundial de computadores e a rede local. Caberá ao nosso futuro NGAV, executado no *proxy*, monitorar o tráfego de rede. Dessa forma, será minimizada a carga de trabalho do nosso NGAV quando executado no computador pessoal. Para tal, faz-se necessária a criação de uma nova *Web-Server Next Generation Sandbox* dotada de uma arquitetura composta por um servidor web, um servidor *proxy* e múltiplos computadores pessoais. O objetivo futuro da nossa NVAG é suprir as limitações dos mecanismos de defesa dos *proxies* os quais são baseados em listas negras assim com os antivírus comerciais. Logo, não será mais necessário aguardar que a rede local seja infectada e, em sequência haja a denuncia de comportamentos anômalos para, então, tomar-se providências quanto à detecção de um novo servidor web malicioso.

Referências

AMOR, N. B.; BENFERHAT, S.; ELOUEDI, Z. Naive bayes vs decision trees in intrusion detection systems., In Proceedings of the 2004 ACM symposium on Applied computing, pág. 420–424., 2004.

AZEVEDO, W.; LIMA, S.; FERNANDES, I.; ROCHA, A. . F. R.; SILVA-FILHO, A..; SANTOS, W. Fuzzy Morphological Extreme Learning Machines to detect and classify masses in mammograms, IEEE International Conference on Fuzzy Systems (FUZZIEEE), Istanbul., 2015.

AZEVEDO, W.; LIMA, S.; FERNANDES, I.; ROCHA, A. . F. R.; SILVA-FILHO, A.; SANTOS, W. Morphological extreme learning machines applied to detect and classify masses in mammograms, International Joint Conference on Neural Networks (IJCNN), Killarney., 2015b.

CONRAD, E.; MISENAR, S.; FELDMAN, J., Eleventh Hour CISSP (Certified Information Systems Security Professional). Elsevier, 2017.

CORDEIRO, F. R.; LIMA, S. M. L.; SILVA-FILHO; SANTOS, W. P. Segmentation of Mammography by Applying Extreme Learning Machine in Tumor Detection, In: International Conference on Intelligent Data Engineering and Automated Learning (IDEAL), 2012, Natal., 2012.

HUANG, G. B.; ZHOU, H.; DING, X. E.; ZHANG, R. Extreme Learning Machine for Regression and MultiClass Classification. IEEE Transactions on Systems, Man, and Cybernetics. Volume 42, número 2, pág. 513-529, 2012.

INTEL. McAfee Labs: Threat Report. Threats Statistics: Malware, Incidents, Web and Network Threats., 2018.

JAYASINGHE, G.; S., C. J.; BERTOK, P. Efficient and effective realtime prediction of drive-by download attacks. , Journal of Network and Computer Applications 38 (2014) 135–149, 2014.

KAPLAN, S.; SIEFERT, C. NOFUS: Automatically Detecting" + String.fromCharCode(32) + "ObFuSCateD".toLowerCase() + "JavaScript Code"., Microsoft Research Technical Report. MSR-TR-2011-57., 2013.

LIMA, S. M. L.; SILVA, H. K. D. L.; LUZ, J. H. D. S.; SILVA, S. L. D. P.; LIMA, H. J. D. N.; ANDRADE, A. B. A. D.; SILVA., A. M. D. Antivírus dotado de Rede Neural Artificial visandoDetectar MalwaresPreventivamente, iSys: Revista Brasileira de Sistemas de Informação (Brazilian Journal of Information Systems), 11(4), 31-62., 2018.

LIMA, S. M. L.; SILVA-FILHO, A. G.; SANTOS, W. P. A methodology for classification of lesions in mammographies using Zernike Moments, ELM and SVM Neural Networks in a multi-kernel approach , In: 2014 IEEE International Conference on Systems, Man and Cybernetics SMC, San Diego., 2014.

LIMA, S. M. L.; SILVA-FILHO, A. G.; SANTOS, W. P. Detection and classification of masses in mammographic images in a multi-kernel approach., Computer Methods and Programs in Biomedicine. 134, (2016), 11-29., 2016.

MICROSOFT. Microsoft Computing Safety Index WorldWide Report, Disponível em: https://news.microsoft.com/pt-br/microsoft-lanca-o-indice-de-cidadania-digital-e-incentiva-as-pessoas-a-ter-mais-empatia-online/. Acesso em Junho de 2019., 2017.

MINNESOTA/USA. University of Minnesota/USA Research, Disponível em: http://www1.umn.edu/news/news-releases/2008/UR_RELEASE_MIG_4855.html. Acesso em Junho de 2019, 2008.

PAEMAL. PHP Analysis Environment Applied to Malware Machine Learning, Disponível em: https://github.com/rewema/paemal. Acesso em Junho de 2019., 2019.

PALOALTO. The network security company. The Modern Malware Review. Analysis of New and Evasive Malware in Live Enterprise Networks, Primeira edição, 2013.

PEKTAS, A.; ACARMAN, T. Classification of Malware Families based on Runtime Behaviors. , Journal of Information Security and Applications 37 (2017) 91-100., 2017.

SANS. SANS Institute InfoSec Reading Room. Out with The Old, In with The New: Replacing Traditional Antivirus, Disponível em: https://www.sans.org/reading-room/whitepapers/analyst/old-new-replacing-traditional-antivirus-37377. Acesso em Junho de 2019., 2019.

SESHAGIRI, P.; VAZHAYIL, A.; SRIRAM, P. AMA: Static Code Analysis of Web Page for the Detection of Malicious Scripts. , Procedia Computer Science, Volume 93, 768-773, 2016.

SKYBOX. Skybox Security Vulnerability and Threat Trends Report 2018.Analysis of current vulnerabilities, exploits and threats in play., Disponível em: https://lp.skyboxsecurity.com/rs/skyboxsecurity/images/Skybox_Report_Vulnerability_Threat_Trends_18.pdf. Acesso em Junho de 2019, 2018.

SKYCURE. Skycure Mobile Threat Defense. Mobile Threat Intelligence Report Q1 2016, Disponível em: https://www.symantec.com/content/dam/symantec/docs/reports/skycure-mobile-threat-intelligence-report-q1-2016-en.pdf. Acesso em Junho de 2019, 2016.

SOPHOS. Sophos Security made simple. Security Threat Report 2014. Smarter, Shadier, Stealthier Malware, Disponível em: https://www.sophos.com/en-us/medialibrary/pdfs/other/sophos-security-threat-report-2014.pdf. Acesso em Junho de 2019, 2014.

SYMANTEC. Symantec Reports. Internet Security Threat Report: 2001 Trends, Volume 17, Symantec Corporation., 2012.

VIRUS. VirusShare: malware files database, Disponível em: https://virusshare.com. Acesso em junho de 2019, 2019.

VIRUSSHARE. VirusShare: malware files database, Disponível em: https://virusshare.com. Acessado em Junho de 2019, 2019.

VIRUSTOTAL. Online service in order to identify malware files by main commercial antiviruses worldwide, Disponível em: https://www.virustotal.com. Acesso em Fevereiro de 2019., 2019.

WANG, Y.; QIU, Y.; THAI, T.; MOORE, K.; LIU, H.; B, Z. A two-step convolutional neural network based computer-aided detection scheme for automatically segmenting adipose tissue volume depicting on CT images., Computer Methods and Programs Biomedicine. 144:97-104, 2017.