Submission date: 28/07/2017
Resubmission date: 14/11/2018
Camera ready submission: 03/02/2019

Section: regular article

1st round notification: 26/10/2018
2nd round notification: 22/01/2019
Available online: 14/02/2019
Publication date: 04/04/2019

Antivírus dotado de Rede Neural Artificial visando Detectar *Malwares* Preventivamente

Title: Antivirus endowed with Artificial Neural Network in order to Detect Malwares Preventively

Sidney M. L. Lima [1], Heverton K. de L. Silva [2], João H. da S. Luz [2], Samuel L. de P. Silva [3], Hercília J. do N. Lima [3], Anna B. A. de Andrade [3], Alisson M. da Silva[3]

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

[2] Empresa Bidweb Security IT – Recife, Brasil

[3] UniSãoMiguel, Curso de Segurança da Informação – Recife, Brasil

smll@ecomp.poli.br, {hevertonkleidson155, joaohenriqu4, samuel1d2013, herciliajuliana99, beatrizaugusta25, alisson_marek}@gmail.com

Abstract. *Malware is a software whose main goal is to access an device without the explicit permission of its owner. Antivirus is the most popular mechanism in order to detect malwares. Commercial antiviruses have as strategy to wait for a user to be infected and in sequence to report an anomalous behavior of her/his device, so that the antivirus manufacturer can take action. Then, in order to overcome the limitations of commercial antivirus, the proposed work creates an antivirus capable of preemptively identifying the modus operandi of a malicious application. The proposed antivirus employs artificial intelligence and can detect a malicious application even before it is run by the user. On average, the smart antivirus created can distinguish malware applications from benign ones in 98.13% of cases.*

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

Resumo. *O malware é um software o qual tem como principal objetivo acessar um dispositivo alheio sem permissão explícita de seu proprietário. Os antivírus constituem o mecanismo mais popular quanto à detecção de malwares. Os antivírus comerciais têm como estratégia aguardar que algum usuário seja infectado e em sequência denuncie um comportamento anômalo de seu dispositivo, para que, então, a fabricante do antivírus possa tomar providências. Então, visando suprir as limitações dos antivírus comerciais, o trabalho proposto cria um antivírus capaz de identificar, preventivamente, o modus operandi de uma aplicação maliciosa. O antivírus proposto emprega inteligência artificial e consegue detectar um aplicativo mal-intencionado antes mesmo dele ser executado pelo usuário. Em média, o antivírus inteligente criado consegue distinguir os aplicativos malwares dos benignos em 98,13% dos casos.*

Cite as: *Lima, S. M. L., Silva, H. K. L., Luz, J. H. S., Silva, S. L. P., Lima, H. J. H.N., Andrade, A. B. A. & Silva, A. M. (2018). Antivirus endowed with Artificial Neural Network in order to Detect Malwares Preventively (Antivírus dotado de Rede Neural Artificial visando Detectar Malwares Preventivamente). iSys: Revista Brasileira de Sistemas de Informação (Brazilian Journal of Information Systems), 11(4), 31-62.*

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

1. Introdução

"*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. Segundo o Centro de Estudos, Respostas e Tratamento de Segurança (CERT.BR, 2016), de 2013 para 2014, o número de notificações de *cyber*-ataques reportadas à entidade aumentou 197%. Aumentos semelhantes a esse tem se repetido em uma sequência de anos seguidos. Com a crescente popularização da internet, a tendência é de que esses números continuem crescendo de forma rápida, ainda durante alguns anos, visto que a internet é o principal meio de propagação de aplicações maliciosas (CERT.BR, 2016).

Sem dúvidas, os antivírus constituem o mecanismo mais popular quanto à segurança de informação. Eles estão presentes em 95% dos computadores pessoais e são associados ao combate de aplicações mal intencionadas (MICROSOFT, 2014). Tecnicamente, o vírus é apenas uma categoria dentre outras quanto a *malwares* como *worm*, *backdoor*, cavalo de troia, *spyware*, *rootkits*, *ransomware*, entre outros. Ao invés de *antimalware*, o termo antivírus costuma ser aplicado para definir ferramentas computacionais que visam prevenir, detectar e eliminar os *malwares* e seus malefícios.

O *modus operandi* dos antivírus comerciais é majoritariamente a identificação do executável malicioso com base em assinaturas (FILHO, *et al*., 2011). Isso quer dizer, o executável suspeito tem seu nome comparado a uma lista negra confeccionada a partir de denúncias prévias. Além disso, alguns antivírus comerciais dividem esse executável suspeito em pequenas porções (*chunks*). Assim, se um ou mais *chunks* do executável suspeito estiver presente na base de dados do antivírus, logo ele é classificado como *malware* (FILHO, *et al*., 2011). Cabe ressaltar que a comparação entre *chunks* da aplicação suspeita e a base de dados do antivírus pode se tornar inviável visto que são criados milhões de *malwares* anualmente (INTEL, 2017). Logo, averiguar a presença de partes de um executável suspeito em todos os milhões de exemplares maliciosos, possivelmente, é um problema computacional inviável a um usuário comum visto que essa tarefa poderia durar dias ou meses.

Outro grande problema dessa estratégia, adotada pelos antivírus comerciais, é para que haja a detecção de uma nova praga virtual é requerido que algumas máquinas já tenham sido infectadas. Cabe ressaltar que não basta apenas a detecção e eliminação do executável malicioso para que a vítima esteja livre de sua atuação. Além da eliminação do *malware*, é necessário desfazer todas as suas malfeitorias como, por exemplo, ter desabilitado os mecanismos de defesa da vítima, inclui-se *firewall*, *plugins* de segurança e os próprios antivírus. Logo, não é apropriada a estratégia de aguardar que uma vítima seja infectada e, em sequência denuncie um comportamento anômalo de seu dispositivo, para, então, tomar-se providências quanto à detecção de um novo *malware*. Enfatiza-se que uma única pessoa ou instituição infectada pode ter prejuízos irrecuperáveis.

Inclui-se como adversidade, no combate a aplicações mal intencionadas, o fato dos antivírus comerciais não possuírem um padrão na classificação dos *malwares* (FILHO, *et al*., 2011). Dessa forma, o tempo em que as fabricantes reagem a uma nova praga é afetado drasticamente. Como não existe um padrão, os antivírus dão os nomes

que desejam, por exemplo, uma empresa pode categorizar um *malware* como "Malware.1" e uma segunda empresa classificá-lo como "Malware12310". Logo, a falta de um padrão, além do não compartilhamento de informações entre as fabricantes de antivírus, dificulta a detecção rápida e eficaz de uma aplicação mal intencionada.

Dadas as deficiências dos antivírus convencionais, o trabalho proposto investiga (*i*) 86 antivírus comerciais. A detecção de malwares variou entre 0% a 99,11%, a depender do antivírus avaliado. Em média, houve a detecção 54,84% das pragas virtuais. Com aspecto desfavorável, os antivírus, em média, atestaram falsos negativos e foram omissos em 14,34% e 30,82% dos casos, respectivamente. Além disso, cerca de 17% dos antivírus não foram capazes de diagnosticar qualquer uma das amostras maliciosas. Enfatiza-se que os malwares analisados, no trabalho proposto, são de domínio público e têm as suas atuações maliciosas largamente documentadas pelos institutos de *cyber*-pesquisas (REWEMA, 2019). Mesmo assim, quase a quinta parte dos antivírus comerciais avaliados não tinham qualquer conhecimento sobre as existências dos malwares investigados. Nota-se, a limitação dos antivírus tradicionais quanto à robustez de serviços em tempo real e em larga escala.

De modo a suprir as limitações do antivírus comerciais, o estado-da-arte propõe extrair características do arquivo, de maneira preventiva, antes de executá-lo. O executável passa por um processo de *disassembling* visando reverter o arquivo binário em seu código fonte. Logo, o algoritmo, referente ao executável, pode ser estudado e, portanto, é possível investigar a intenção maliciosa do arquivo suspeito (ADKINS, *et al*., 2013), (ALAZAB, *et al*., 2015), (KHODAMORADI, *et al*., 2015), (FAN, *et al*., 2016). Então, a extração de características do algoritmo, referente ao arquivo periciado, possibilita a definição de uma sequencia de eventos suspeitos. Dessa forma, é possível determinar a intenção maliciosa do arquivo antes mesmo dele ser executado pelo usuário. Através da análise do algoritmo, o estado-da-arte investiga se o arquivo auditado cria, exclui, altera e faz o *download* de outros arquivos pela internet. Além disso, o estudo do algoritmo torna viável o rastreamento preventivo do tráfego de rede provocado pelo arquivo suspeito.

Após a extração das características dos executáveis suspeitos, o estado-da-arte costuma empregar modelos heurísticos onde são atribuídos pesos ponderados, de maneira empírica, às características dos arquivos auditados. A meta é agrupar os aplicativos suspeitos em duas classes: benigno e *malwares*. PRADO, *et al*. (2016), por exemplo, atribui pesos de 1 a 5 a comportamentos empiricamente classificados como suspeitos. Na obra de PRADO, *et al*. (2016), há a atribuição do peso 2, de forma heurística, caso o arquivo auditado tente criar um novo arquivo visto que esse comportamento pode estar atrelado a pragas virtuais. Modelos empíricos e heurísticos conseguem bons resultados mesmo sendo baseados na intuição do analista. Como efeito colateral, o empirismo pode provocar algumas distorções como, por exemplo, uma grande quantidade de falsos positivos. Ao empregar o empirismo, aproximadamente 1 em cada 5 executáveis benignos são equivocadamente classificados como *malware* (PRADO, *et al*., 2016).

As limitações de modelos heurísticos podem ser supridas por técnicas de inteligência artificial baseadas em aprendizado de máquina. Então, ao invés da intuição do analista, o peso ponderado referente a um comportamento suspeito é determinado através de máquinas de aprendizado estatístico. Inteligência artificial consegue automatizar de forma inteligente muitas tarefas, analisando milhares de arquivos,

extraindo características deles e os ponderando estatisticamente. A inteligência artificial tem sido largamente aplicada nas mais diversas áreas, porém segurança da informação não é uma delas (HENKE, *et al*., 2011). Inteligência artificial pode contribuir bastante para o avanço da segurança em dispositivos. Há iniciativas, mas ainda se encontram em um estágio inicial (HENKE, *et al*., 2011).

Visando suprir as limitações e imprecisões dos modelos empíricos e heurísticos, o projeto proposto emprega redes neurais como técnica de inteligência artificial baseada em máquinas de aprendizado estatístico. Logo, o trabalho proposto tem como meta identificar, de forma estatística, comportamentos previamente classificados como suspeitos em um tempo de resposta viável a um usuário comum. Então, não é necessário aguardar meses ou anos até que o executável suspeito seja denunciado por um cliente assim como os antivírus comerciais atuam.

Quanto ao ambiente experimental, o trabalho proposto cria *(ii)* a REWEMA (***Re****trieval of 32-bit* ***W****indows Architecture* ***E****xecutables Applied to* ***M****alware* ***A****nalysis* – Redistribuição de Executáveis do Windows em Arquiteturas de 32-bits Aplicados a Análise de Malware) (REWEMA, 2019). Por disputas comerciais, as fabricantes dos antivírus não compartilham, entre elas, as suas respectivas listas negras de *malwares*. A base de dados REWEMA é empregada como repositório do antivírus inteligente criado. A REWEMA apresenta 3136 executáveis malignos, visando arquiteturas de 32 *bits*, com suas respectivas análises emitidas pelo antivírus comerciais extraídas pela plataforma VirusTotal. A REWEMA disponibiliza outros 3136 executáveis benignos os quais, em conjunto com as amostragens malignas, são empregados no aprendizado através de redes neurais artificiais.

No tocante aos experimentos, os executáveis são divididos em duas classes: benigno e *malwares*. Na etapa de extração de características, são analisados 630 comportamentos e rotinas suspeitas extraídas do algoritmo pertencentes ao arquivo auditado. Na etapa de classificação, há o reconhecimento de padrão dos comportamentos atrelados a atividades maliciosas, especificamente, através de Redes Neurais Artificiais MLP (*Multilayer Perceptron* - Perceptron com Múltiplas Camadas) (LIMA, *et al*., 2016). As características extraídas dos executáveis servem como atributos de entrada das redes neurais artificiais no intuito de tornar o antivírus inteligente capaz de reconhecer atividades maliciosas antes mesmo de ser executado pelo usuário. O trabalho proposto e alcança um desempenho médio de 98,13% na distinção entre executáveis benignos e *malwares*, acompanhado de um tempo de resposta médio de apenas 0,06 segundos.

2. Limitação 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, 2018). 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 programas (*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, 2018).

Por intermédio da plataforma VirusTotal[1], o trabalho proposto investiga 86 antivírus comerciais com seus respectivos resultados apresentados da Tabela 1 à Tabela 4. Foram empregados 3136 executáveis binários maliciosos obtidos da base de dados autoral nomeada REWEMA (REWEMA, 2019). O objetivo do trabalho 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 *malwares*, 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 malwares são enviados ao servidor pertencente à plataforma VirusTotal. Após isso, os malwares são analisados pelos 86 antivírus comerciais vinculados ao VirusTotal. Logo, os antivírus provêm seus diagnósticos para os malwares submetidos à plataforma. Conforme ilustra a Figura 1, o VirusTotal permite a possibilidade de emissão de três tipos diferentes de diagnósticos; *malware*, benigno e omissão.

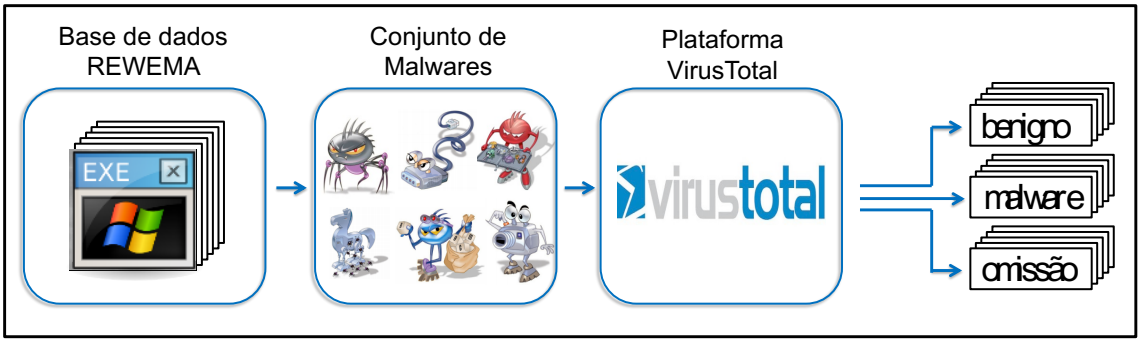

Figura 1. Diagrama da API VirusTotal.

Quanto à primeira possibilidade do VirusTotal, o antivírus detecta a malignidade do arquivo suspeito. No ambiente experimental proposto, todos os arquivos submetidos são *malwares* documentados por institutos de cyber-pesquisa (REWEMA, 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.

[1] VirusTotal: Serviço online quanto à identificação de malwares pelos principais antivírus comerciais mundiais. Disponível em: https://www.virustotal.com. Acesso em julho de 2018.

Da Tabela 1 à Tabela 4, há a exibição dos resultados alcançados pelos 86 antivírus comerciais avaliados. O antivírus McAfee-GW-Edition obteve o melhor desempenho sendo capaz de detectar 99.11% dos malwares investigados. Uma grande adversidade, no combate a aplicações mal intencionadas, é o fato das fabricantes dos antivírus não compartilharem, entre elas, as suas respectivas listas negras de *malwares* devido a disputas comerciais. Através da análise da Tabela 1 à Tabela 4, o trabalho proposto aponta para um agravante dessa adversidade; uma mesma fabricante de antivírus sequer compartilha as suas bases de dados entre seus distintos antivírus. Observe, por exemplo, que os antivírus McAfee-GW-Edition e McAfee pertencem a uma mesma empresa. As suas listas negras, apesar de robustas, não são compartilhadas entre si. Logo, as estratégias comerciais, de uma mesma empresa, atrapalham o enfretamento a *malwares*. Complementa-se que as fabricantes de antivírus não estão necessariamente preocupadas em evitar as cyber-invasões, e sim em otimizar seus rendimentos comerciais.

A detecção de *malwares* variou entre 0% e 99,11%, a depender do antivírus investigado. Em média, os 86 antivírus foram capazes de detectar 54,84% das pragas virtuais avaliadas, com desvio padrão de 39,67. 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 14,34% dos casos, com desvio padrão de 21,67. 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 17% dos antivírus não emitiram opinião em qualquer uma das 3136 amostras maliciosas. Em média, os antivírus foram omissos em 30,82% dos casos, com desvio padrão de 40,97. A omissão do diagnóstico aponta a limitação dos antivírus quanto à detecção de *malwares* em tempo real.

Inclui-se como adversidade, no combate a aplicações mal intencionadas, o fato dos antivírus comerciais não possuírem um padrão na classificação dos *malwares* como visto da Tabela 5 à Tabela 10. Então, foram escolhidos 3 dos 3136 malwares 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, uma empresa pode categorizar um *malware* como “Malware.1” e uma segunda empresa classificá-lo como “Malware12310”. Logo, a falta de um padrão atrapalha as estratégias de *cyber*-vigilância visto que cada categoria de malware deveria ter tratamentos (vacinas) distintos. É importante ressaltar que uma mesma fabricante de antivírus pode não compartilhar, entre seus produtos, as suas bases de dados e também as suas categorias de malwares. Observe, por exemplo, que os antivírus McAfee-GW-Edition e McAfee pertencem a uma mesma empresa. Como efeito adverso, um mesmo malware pode ser classificado como “BehavesLike.Win32.Downloader.lc” e “W32/Sdbot.gen.an” pelo McAfee-GW-Edition e McAfee, respectivamente. Enfatiza-se que, no site oficial da fabricante McAfee, não há qualquer documentação sobre tais categorias de malwares. Conclui-se que, devido às estratégias comerciais, as fabricantes de antivírus atrapalham as estratégias de cyber-proteção.

Ressalta-se que é inviável o aprendizado de máquina supervisionado visando reconhecimento de padrão de categorias de malwares. Devido a esse emaranhado confuso, visto da Tabela 5 à Tabela 10, é estatisticamente improvável que alguma técnica

de aprendizado de máquina adquira capacidade de generalização. O aprendizado supervisionado deveria prover classificações de milhares de multi-classes (*labels*) providas pelos especialistas (antivírus).

Tabela 1. Parte 1: Resultado dos 86 antivírus comerciais.

Antivírus	Detecção (%)	Falso negativo (%)	Omissão (%)
McAfee-GW-Edition	**99,11%**	0,89%	0,00%
McAfee	98,98%	1,02%	0,00%
Panda	98,76%	1,15%	0,10%
Comodo	98,37%	1,59%	0,03%
Kaspersky	98,34%	1,56%	0,10%
NANO-Antivirus	98,25%	1,69%	0,06%
Symantec	97,64%	2,26%	0,10%
GData	97,23%	2,77%	0,51%
BitDefender	97,23%	2,77%	0,00%
MicroWorld-eScan	96,94%	2,74%	0,32%
F-Secure	98,76%	3,00%	1,12%
Qihoo-360	95,66%	1,69%	2,65%
VIPRE	95,54%	4,46%	0,00%
Sophos	95,38%	4,53%	0,10%
Emsisoft	95,22%	4,21%	0,57%
Avira	95,18%	2,39%	2,42%
CMC	95,15%	4,37%	0,48%
Arcabit	94,58%	4,21%	1,21%
DrWeb	94,36%	5,61%	0,03%
Ad-Aware	93,88%	3,32%	2,81%
Fortinet	93,85%	6,15%	0,00%
Rising	93,72%	5,01%	1,28%
Antiy-AVL	92,38%	7,21%	0,41%
Jiangmin	91,87%	8,04%	0,10%
Ikarus	90,50%	8,20%	1,31%

Tabela 2. Parte 2: Resultado dos 86 antivírus comerciais.

Antivírus	Detecção (%)	Falso negativo (%)	Omissão (%)
Cyren	90,34%	8,67%	0,99%
Zillya	89,89%	8,32%	1,79%
F-Prot	89,51%	10,49%	0,00%
TheHacker	87,69%	12,28%	0,03%
Avast	86,19%	13,71%	0,10%
ESET-NOD32	86,13%	13,87%	0,00%
AVG	85,55%	14,45%	0,00%
Microsoft	85,20%	14,80%	0,00%
AegisLab	82,40%	17,12%	0,48%
Tencent	82,21%	6,44%	11,35%
VBA32	81,12%	18,88%	0,00%
TrendMicro-HouseCal	79,02%	17,83%	3,16%
K7AntiVirus	78,95%	21,05%	0,00%
K7GW	78,83%	21,17%	0,00%
AVware	78,64%	4,02%	17,25%
TrendMicro	78,54%	18,53%	2,93%
ALYac	77,20%	16,55%	6,25%
Yandex	70,22%	3,19%	26,59%
AhnLab-V3	69,29%	30,52%	0,19%
nProtect	67,22%	32,49%	0,29%
ViRobot	64,06%	35,94%	0,00%
ClamAV	59,41%	40,31%	0,29%
ZoneAlarm	54,37%	1,02%	44,61%
Webroot	52,04%	2,20%	45,76%
MAX	50,77%	1,66%	47,58%
Cylance	48,60%	2,87%	48,53%
TotalDefense	41,04%	53,13%	5,84%
CrowdStrike	40,88%	21,78%	37,34%

Tabela 3. Parte 3: Resultado dos 86 antivírus comerciais.

Antivírus	Detecção (%)	Falso negativo (%)	Omissão (%)
Invincea	40,27%	27,04%	32,68%
Endgame	40,18%	14,99%	44,83%
Baidu	39,48%	34,98%	25,54%
CAT-QuickHeal	39,19%	60,81%	0,00%
Agnitum	25,22%	0,80%	73,98%
Bkav	22,39%	73,02%	4,59%
SentinelOne	20,92%	32,49%	46,59%
Kingsoft	19,87%	57,59%	22,54%
Paloalto	17,60%	37,12%	45,28%
SUPERAntiSpyware	7,33%	92,67%	0,00%
Malwarebytes	7,21%	92,19%	0,61%
1ByteHero	2,36%	23,50%	74,14%
Zoner	2,17%	96,65%	1,18%
Norman	1,18%	0,03%	98,79%
AntiVir	0,99%	0,00%	99,01%
Commtouch	0,89%	0,10%	99,01%
Ahnlab	0,03%	0,00%	99,97%
Alibaba	0,00%	35,49%	64,51%
VirusBuster	0,00%	0,00%	100,00%
NOD32	0,00%	0,00%	100,00%
eSafe	0,00%	0,00%	100,00%
eTrust-Vet	0,00%	0,00%	100,00%
Authentium	0,00%	0,00%	100,00%
Prevx	0,00%	0,00%	100,00%
Sunbelt	0,00%	0,00%	100,00%
PCTools	0,00%	0,00%	100,00%
Squared	0,00%	0,00%	100,00%
WhiteArmor	0,00%	0,00%	100,00%

Tabela 4. Parte 4: Resultado dos 86 antivírus comerciais.

Antivírus	Detecção (%)	Falso negativo (%)	Omissão (%)
Command	0,00%	0,00%	100,00%
SAVMail	0,00%	0,00%	100,00%
FileAdvisor	0,00%	0,00%	100,00%
Ewido	0,00%	0,00%	100,00%
Webwasher-Gateway	0,00%	0,00%	100,00%

Tabela 5. Parte 1: Miscelânea de classificações providas pelos antivírus comerciais.

Antivírus	Backdoor.IRC.exe	AndroRat Binder_Patched.exe	Rustock.C.exe
McAfee-GW-Edition	BehavesLike.Win32.Downloader.lc	RDN/Generic.grp	BehavesLike.Win32.Ramnit.kc
McAfee	W32/Sdbot.gen.an	RDN/Generic.grp	Generic.dx!FDAFB3A14338
Panda	Trj/Genetic.gen	Trj/GdSda.A	Trj/Agent.EDT
Comodo	Backdoor.IRC.Bot-gen	UnclassifiedMalware	Trojan-Clicker.Win32.Costrat.bk
Kaspersky	Backdoor.Win32.IRCBot.gen	UDS:DangerousObject.Multi.Generic	Trojan-Clicker.Win32.Costrat.bk
NANO-Antivirus	Trojan.Win32.IRCBot.dkptuu	Riskware.Win32.Androrat.ebhzgr	Trojan.Win32.Spambot.eatxnv
Symantec	W32.IRCBot	Trojan.Gen.2	Backdoor.Rustock.B
GData	Generic.Malware.SIBdldg.28B98B92	Trojan.GenericKD.3407708	Backdoor.Rustock.Gen.1
BitDefender	Generic.Malware.SIBdldg.28B98B92	Trojan.GenericKD.3407708	Backdoor.Rustock.Gen.1
MicroWorld-eScan	Generic.Malware.SIBdldg.28B98B92	Trojan.GenericKD.3407708	Backdoor.Rustock.Gen.1
F-Secure	Generic.Malware.SIBdldg.28B98B92	Trojan.GenericKD.3407708	Backdoor.Rustock.Gen.1
Qihoo-360	Omission	Win32/Trojan.125	Win32/Trojan.Clicker.89e
VIPRE	Trojan.Win32.Ircbot!cobra (v)	Trojan.Win32.Generic!BT	Trojan.Win32.Generic!BT
Sophos	W32/Sdbot-Gen	Generic PUA BL (PUA)	Mal/RKRustok-A
Emsisoft	Generic.Malware.SIBdldg.28B98B92 (B)	Trojan.GenericKD.3407708 (B)	Backdoor.Rustock.Gen.1 (B)
Avira	WORM/SdBot.13824	TR/NetSeal.575488	TR/Dropper.Gen

Tabela 6. Parte 2: Miscelânea de classificações providas pelos antivírus comerciais.

Antivírus	Backdoor.IRC.exe	AndroRat Binder_Patched.exe	Rustock.C.exe
CMC	Generic.Win32.fee5446b04!MD	Benign	Generic.Win32.56c37100ce!MD
Arcabit	Generic.Malware.SIBdldg.28B98B92	Trojan.Generic.D33FF5C',	Backdoor.Rustock.Gen.1
DrWeb	Win32.IRC.Bot.based	Tool.Androrat	Trojan.Spambot
Ad-Aware	Generic.Malware.SIBdldg.28B98B92	Trojan.GenericKD.3407708	Backdoor.Rustock.Gen.1
Fortinet	W32/SDBot.ABC!tr.bdr	MSIL/Generic.DN.1135A0!tr	W32/Generic.CON!tr
Rising	PE:Malware.Generic(Thunder)!1.A1C4 [F]	Trojan.Generic (cloud:rtYiZ8N6s3M)	Backdoor.Rustock.i (classic)
Antiy-AVL	Trojan[Backdoor]/IRC.Bot-gen	Benign	Trojan/Win32.SGeneric
Jiangmin	Backdoor/IRC.Bot-gen	Benign	TrojanClicker.Costrat.lx
Ikarus	Backdoor.Win32.SdBot	PUA.MSIL.NetSeal	Backdoor.WinNT.Rustock
Cyren	W32/Bloop.A.gen!Eldorado	Benign	W32/Rustock.CQKI-9104
Zillya	Backdoor.Bot.Win32.15	Adware.MegaSearch.Win32.17554	Trojan.Castrat.Win32.62
F-Prot	W32/Bloop.A.gen!Eldorado	Benign	W32/Rustock.C'
TheHacker	IRC/SdBot	Benign	Trojan/Clicker.Costrat.bk
Avast	Win32:SdBot-gen17 [Trj]	Win32:Malware-gen	Win32:Susn-F [Trj]
ESET-NOD32	IRC.Bot-gen	A variant of MSIL/Packed.NetSeal.A suspicious	a variant of Win32/Rootkit.Kryptik.BP
AVG	IRC/BackDoor.SdBot.MP	Win32:Malware-gen	Crypt5.ANAA
Microsoft	Backdoor:Win32/Sdbot	MonitoringTool:AndroidOS/AndroRat	Backdoor:WinNT/Rustock.D
AegisLab	Benign	Gen.Variant.Barys!c	Troj.Clicker.W32.Costrat.bk!c
Tencent	Win32.Backdoor.Ircbot.Eivd	Benign	Win32.Trojan.Costrat.Egyh
VBA32	BScope.Backdoor.Win32.SdBot	Benign	Malware-Cryptor.Win32.015
TrendMicro-HouseCal	BKDR_SDBOT.A	TROJ_GEN.R0C1C0EDK17	BKDR_RUSTOCK.AR

Tabela 7. Parte 3: Miscelânea de classificações providas pelos antivírus comerciais.

Antivírus	Backdoor.IRC.exe	AndroRat Binder_Patched.exe	Rustock.C.exe
Tencent	Win32.Backdoor.Ircbot.Eivd	Benign	Win32.Trojan.Costrat.Egyh
VBA32	BScope.Backdoor.Win32.SdBot	Benign	Malware-Cryptor.Win32.015
TrendMicro-HouseCal	BKDR_SDBOT.A	TROJ_GEN.R0C1C0EDK17	BKDR_RUSTOCK.AR
McAfee-GW-Edition	BehavesLike.Win32.Downloader.lc	RDN/Generic.grp	BehavesLike.Win32.Ramnit.kc
McAfee	W32/Sdbot.gen.an	RDN/Generic.grp	Generic.dx!FDAFB3A14338
Panda	Trj/Genetic.gen	Trj/GdSda.A	Trj/Agent.EDT
Comodo	Backdoor.IRC.Bot-gen	UnclassifiedMalware	Trojan-Clicker.Win32.Costrat.bk
Kaspersky	Backdoor.Win32.IRCBot.gen	UDS:DangerousObject.Multi.Generic	Trojan-Clicker.Win32.Costrat.bk
NANO-Antivirus	Trojan.Win32.IRCBot.dkptuu	Riskware.Win32.Androrat.ebhzgr	Trojan.Win32.Spambot.eatxnv
Symantec	W32.IRCBot	Trojan.Gen.2	Backdoor.Rustock.B
GData	Generic.Malware.SIBdldg.28B98B92	Trojan.GenericKD.3407708	Backdoor.Rustock.Gen.1
BitDefender	Generic.Malware.SIBdldg.28B98B92	Trojan.GenericKD.3407708	Backdoor.Rustock.Gen.1
MicroWorld-eScan	Generic.Malware.SIBdldg.28B98B92	Trojan.GenericKD.3407708	Backdoor.Rustock.Gen.1
F-Secure	Generic.Malware.SIBdldg.28B98B92	Trojan.GenericKD.3407708	Backdoor.Rustock.Gen.1
Qihoo-360	Omission	Win32/Trojan.125	Win32/Trojan.Clicker.89e
VIPRE	Trojan.Win32.Ircbot!cobra (v)	Trojan.Win32.Generic!BT	Trojan.Win32.Generic!BT
Sophos	W32/Sdbot-Gen	Generic PUA BL (PUA)	Mal/RKRustok-A
Emsisoft	Generic.Malware.SIBdldg.28B98B92 (B)	Trojan.GenericKD.3407708 (B)	Backdoor.Rustock.Gen.1 (B)
Avira	WORM/SdBot.13824	TR/NetSeal.575488	TR/Dropper.Gen
CMC	Generic.Win32.fee5446b04!MD	Benign	Generic.Win32.56c37100ce!MD
Arcabit	Generic.Malware.SIBdldg.28B98B92	Trojan.Generic.D33FF5C',	Backdoor.Rustock.Gen.1
DrWeb	Win32.IRC.Bot.based	Tool.Androrat	Trojan.Spambot

Tabela 8. Parte 4: Miscelânea de classificações providas pelos antivírus comerciais.

Antivírus	Backdoor.IRC.exe	AndroRat Binder_Patched.exe	Rustock.C.exe
Ad-Aware	Generic.Malware.SIBdldg.28B98B92	Trojan.GenericKD.3407708	Backdoor.Rustock.Gen.1
Fortinet	W32/SDBot.ABC!tr.bdr	MSIL/Generic.DN.1135A0!tr	W32/Generic.CON!tr
Rising	PE:Malware.Generic(Thunder)!1.A1C4 [F]	Trojan.Generic (cloud:rtYiZ8N6s3M)	Backdoor.Rustock.i (classic)
Antiy-AVL	Trojan[Backdoor]/IRC.Bot-gen	Benign	Trojan/Win32.SGeneric
Jiangmin	Backdoor/IRC.Bot-gen	Benign	TrojanClicker.Costrat.lx
Ikarus	Backdoor.Win32.SdBot	PUA.MSIL.NetSeal	Backdoor.WinNT.Rustock
Cyren	W32/Bloop.A.gen!Eldorado	Benign	W32/Rustock.CQKI-9104
Zillya	Backdoor.Bot.Win32.15	Adware.MegaSearch.Win32.17554	Trojan.Castrat.Win32.62
F-Prot	W32/Bloop.A.gen!Eldorado	Benign	W32/Rustock.C'
TheHacker	IRC/SdBot	Benign	Trojan/Clicker.Costrat.bk
Avast	Win32:SdBot-gen17 [Trj]	Win32:Malware-gen	Win32:Susn-F [Trj]
ESET-NOD32	IRC.Bot-gen	A variant of MSIL/Packed.NetSeal.A suspicious	a variant of Win32/Rootkit.Kryptik.BP
AVG	IRC/BackDoor.SdBot.MP	Win32:Malware-gen	Crypt5.ANAA
Microsoft	Backdoor:Win32/Sdbot	MonitoringTool:AndroidOS/AndroRat	Backdoor:WinNT/Rustock.D
AegisLab	Benign	Gen.Variant.Barys!c	Troj.Clicker.W32.Costrat.bk!c
Tencent	Win32.Backdoor.Ircbot.Eivd	Benign	Win32.Trojan.Costrat.Egyh
VBA32	BScope.Backdoor.Win32.SdBot	Benign	Malware-Cryptor.Win32.015
TrendMicro-HouseCal	BKDR_SDBOT.A	TROJ_GEN.R0C1C0EDK17	BKDR_RUSTOCK.AR
K7AntiVirus	Backdoor (04c514e61)	Trojan (004b4bad1)	RootKit (004d16801)
K7GW	Backdoor (04c514e61)	Trojan (004b4bad1)	RootKit (004d16801)
AVware	Omission	Trojan.Win32.Generic!BT	Trojan.Win32.Generic!BT

Tabela 9. Parte 5: Miscelânea de classificações providas pelos antivírus comerciais.

Antivírus	Backdoor.IRC.exe	AndroRat Binder_Patched.exe	Rustock.C.exe
TrendMicro	BKDR_SDBOT.A	TROJ_GEN.R0C1C0EDK17	Benign
ALYac	Omission	Trojan.GenericKD.3407708	Backdoor.Rustock.Gen.1
Yandex	Omission	Riskware.NetSeal!	Trojan.Rustock!gMcRHMfFn+E
AhnLab-V3	Win32/IRCBot.worm.variant	Benign	Benign
nProtect	Backdoor/W32.Agent.13824.AL	Benign	Trojan-Clicker/W32.Costrat.70570
ViRobot	Backdoor.Win32.Bot-gen.13824[h]	Trojan.Win32.Z.Netseal.575488[h]	Trojan.Win32.Clicker.70570[h]
ClamAV	Trojan.IRCBot.13856	Benign	Win.Trojan.Clicker-1225
ZoneAlarm	Omission	UDS:DangerousObject.Multi.Generic	Omission
Webroot	Omission	W32.Malware.Ml.Vt	W32.Rustock.Rootkit
MAX	Omission	Omission	Omission
Cylance	Omission	Omission	Omission
TotalDefense	Win32/Sdbot!generic	Benign	Benign
CrowdStrike	Omission	malicious_confidence_74% (D)	malicious_confidence_100% (D)
Invincea	Omission	Benign	backdoor.winnt.rustock.d
Endgame	Omission	Benign	malicious (high confidence)
Baidu	Backdoor.Win32.IRCBot.gen	Benign	Win32.Trojan.WisdomEyes.16070401.9500.9995
CAT-QuickHeal	Omission	Monitoringtool.Androrat	Backdoor.Rustock
Agnitum	Backdoor.SdBot.gen	Omission	Omission
Bkav	Benign	Benign	W32.Clodf61.Trojan.3118
SentinelOne	Omission	static engine – malicious	Omission
Kingsoft	Omission	Benign	Benign
Paloalto	Omission	generic.ml	Omission
SUPERAntiSpyware	Benign	Benign	Benign
Malwarebytes	Benign	Benign	Benign
ByteHero	Virus.Win32.Heur.c	Omission	Omission

Tabela 10. Parte 6: Miscelânea de classificações providas pelos antivírus comerciais.

Antivírus	Backdoor.IRC.exe	AndroRat Binder_Patched.exe	Rustock.C.exe
Zoner	Benign	Benign	Benign
Norman	Omission	Omission	Omission
AntiVir	Backdoor (04c514e61)	Omission	RootKit (004d16801)
Commtouch	Omission	Omission	Omission
Ahnlab	Win32/IRCBot.worm.variant	Benign	Benign
Alibaba	Benign	Omission	Omission
VirusBuster	Omission	Omission	Omission
NOD32	Omission	a variant of MSIL/Packed.NetSeal.A suspicious	a variant of Win32/Rootkit.Kryptik.BP
eSafe	Omission	Omission	Omission
eTrust-Vet	Omission	Omission	Omission
Authentium	Omission	Omission	Omission
Prevx	Omission	Omission	Omission
Sunbelt	Omission	Omission	Omission
PCTools	Omission	Omission	Omission
Squared	Omission	Omission	Omission
WhiteArmor	Omission	Omission	Omission
Command	Omission	Omission	Omission
SAVMail	Omission	Omission	Omission
FileAdvisor	Omission	Omission	Omission
Ewido	Omission	Omission	Omission
Webwasher-Gateway	Omission	Omission	Omission

3. Metodologia Proposta

Dada as limitações dos antivírus comerciais, o trabalho proposto visa criar um antivírus, dotado de inteligência artificial, capaz de diferenciar aplicativos malwares de benignos de forma preventiva. A Figura 2 exibe o diagrama da metodologia proposta em diagrama de blocos.

Inicialmente, são extraídas as características do conjunto de executáveis. Logo, as redes neurais artificiais utilizam, como atributos de entrada, essas características extraídas das executáveis os quais são classificados entre benignos e *malwares*. Os resultados da classificação estão descritos no Capítulo 4.

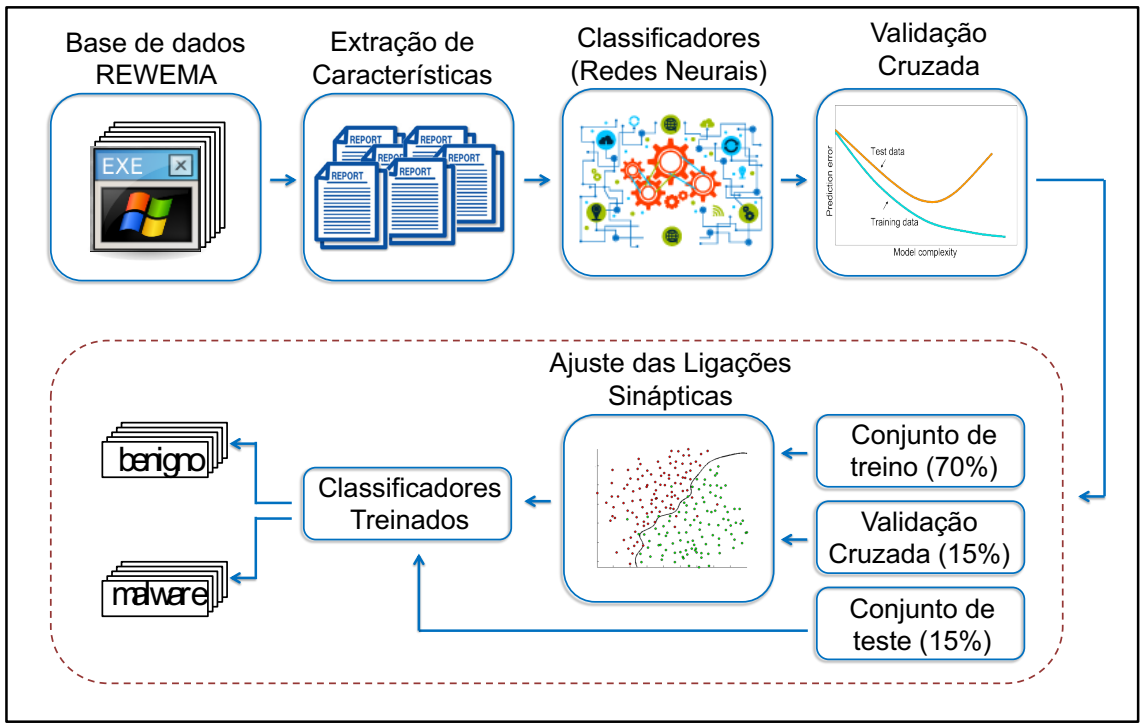

Figura 2. Diagrama da metodologia proposta.

3.1. Materiais e Métodos: Base de Dados REWEMA

O trabalho proposto cria uma base de dados de modo que o antivírus inteligente possa adquirir capacidade de aprendizado mediante as instancias apresentadas. A base de dados criada é batizada de REWEMA (***R**etrieval of 32-bit **W**indows Architecture **E**xecutables Applied to **M**alware **A**nalysis* – Redistribuição de Executáveis do Windows em Arquiteturas de 32-bits Aplicados a Análise de Malware) (REWEMA, 2019). Há 3316 executáveis malignos e outros 3316 executáveis benignos. Logo, a base REWEMA se torna adequada ao aprendizado dotado de inteligência artificial visto que ambas as classes de executáveis apresentam a mesma quantidade.

Quanto aos executáveis maliciosos, o REWEMA é a junção de várias bases de dados de *malwares*. As pragas virtuais foram extraídas de bases de dados disponibilizadas por grupos entusiastas do estudo de *malwares*, como o Vxheaven [2] e TheZoo[3]. Quanto aos executáveis benignos, a aquisição foi oriunda de repositórios de aplicações benignas como Sourceforge [4], Github [5] e Sysinternals [6]. Cabe ressaltar que todos os executáveis benignos foram submetidos ao VirusTotal e todos tiveram a sua benignidade atestada pelos principais antivírus comerciais mundiais. Os diagnósticos, providos pelos

[2] Vxheaven: bases de dados quanto a executáveis malwares em arquiteturas de 32 bits. Disponível em: http://vxheaven.org/. Acesso em junho de 2017.
[3] TheZoo: bases de dados quanto a executáveis malwares em arquiteturas de 32 bits. Disponível em: https://github.com/ytisf/theZoo. Acesso em junho de 2018.
[4] Sourceforge: Repositório de aplicativos benignos. Disponível em: https://sourceforge.net/. Acesso em junho de 2018.
[5] Github: Repositório de aplicativos benignos Disponível em: https://github.com/. Acesso em junho de 2018.
[6] Sysinternals: Repositório de aplicativos benignos. Disponível em: https://live.sysinternals.com/. Acesso em junho de 2018.

VirusTotal, correspondentes aos executáveis benignos e *malwares* estão disponibilizados no endereço virtual da base de dados REWEMA (REWEMA, 2019).

O objetivo da criação da base de dados REWEMA é dar total possibilidade da metodologia proposta ser replicada, por terceiros, em trabalhos futuros. Logo, o artigo proposto, ao disponibilizar, livremente, a sua base de dados viabiliza transparência e imparcialidade à pesquisa, além de demostrar a veracidade dos resultados alcançados. Então, espera-se que a metodologia, a ser relatada no capítulo seguinte, sirva de base para a criação de novos trabalhos científicos.

3.2. Extração de Características dos Executáveis

A extração de características dos executáveis passa pelo processo de *disassembling* visando reverter o arquivo binário em seu código fonte. Logo, o algoritmo, referente ao executável, pode ser estudado e posteriormente classificado pelas redes neurais descritas na próxima seção. Da Tabela 11 à Tabela 13, há as descrições das características às suas respectivas ferramentas, funções e bibliotecas. No total, são extraídas 630 características, de cada executável, referentes aos grupos acima citados. Scripts autorais e a ferramenta *pescanner* [F1] são utilizadas na extração de características dos executáveis. A seguir, são detalhados os grupos de características referentes ao algoritmo dos arquivos investigados.

- ✓ Histograma das instruções responsáveis pela aquisição de dados (*imports*) [F2].
- ✓ Quantidade de sub-rotinas que invocam o TLS (Transport Layer Security).
- ✓ Quantidade de sub-rotinas responsáveis pela exportação de dados (*exports*).
- ✓ Histograma das importações das API (*Application Programming Interface* - Interface de Programação de Aplicações) empregadas pelo executável.
- ✓ Características relacionadas a indícios que o computador tenha sofrido fragmentação no seu disco rígido, além de tentativas de inicialização inválidas acumuladas [F3].
- ✓ Modo de execução do aplicativo. Há duas opções:
  - *softwares* dotados de interface gráfica (GUI);
  - *softwares* executados no console.
- ✓ Características relacionadas ao Sistema Operacional. A forense digital averigua se o arquivo testado tenta:
  - identificar o nome do usuário atual do sistema operacional [F4];
  - acessar funções de interface de programação de aplicativo (API) para criar e gerenciar perfis de usuário atual do SO [F5];
  - detectar o número de milissegundos decorridos desde que o sistema foi inicializado [F6];
  - executar uma operação em um arquivo especifico [F7];
  - identificar a versão do Sistema Operacional Windows em uso [F8];
  - monitorar o tráfego de mensagens internas entre os processos do Sistema [F9];
  - alterar as configurações e conteúdos da inicialização (STARTUPINFO) do SO Windows [F10];
  - permitir que aplicativos acessem a funcionalidade fornecida pelo *shell* do sistema operacional, além de alterá-lo [F11];
  - alterar as mensagens de logon na inicialização do SO Windows [F12];

- alterar aplicativos nativos atrelados a caixas de diálogo padrão para abrir e salvar arquivos, escolhendo cor e fonte, dentre outras customizações [F13];
- configurar o licenciamento do Servidor Windows [F14];
- configurar o Windows Server 2003 [F15];
- alterar as configurações de energia do Sistema [F16];
- abrir um processo, serviço ou biblioteca nativa do Sistema Operacional [F17];
- excluir o contexto de certificados vinculados ao Sistema Operacional [F18];
- copiar um arquivo existente para um novo arquivo [F19];
- criar, abrir, excluir ou alterar um arquivo [F20];
- criar e executar novo(s) processo(s) [F21].
- criar novo(s) diretório(s) [F22];
- procurar por arquivo(s) específico(s) [F23];
- criar um objeto de serviço e o adicionar ao banco de dados do gerenciador de controle de um determinado serviço [F24];
- criptografar dados. Tal estratégia é típica de *ransomwares* os quais sequestram os dados da vítima através da criptografia. Para descriptografar os dados, o invasor pede ao usuário um montante monetário para que possa ter de volta todos os seus dados [F25];
- acessar sistemas de arquivos, dispositivos, processos, *threads* e tratamento de erros [F26];
- alterar as propriedades de som e dispositivo de áudio do sistema [F27];
- acessar informações de conteúdo gráfico para monitores, impressoras e outros dispositivos de saída do SO Windows [F28];
- usar e/ou monitorar a porta USB [F29];
- controlar um *driver* de um determinado dispositivo [F30];
- investigar se uma unidade de disco é uma unidade removível, fixa, de CD/DVD-ROM, de RAM ou de rede [F31];

✓ Características relacionadas ao Registro (Regedit) do SO Windows. 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, mesmo após a exclusão do malware, ocorre devido à inserção de entradas (chaves) maliciosas no Regedit. Logo, quando o sistema operacional é inicializado, o *cyber*-ataque recomeça devido à chave mal-intencionada invocar a vulnerabilidade explorada pelo malware (eg: redirecionar a página inicial do Internet Explorer). Logo, o antivírus criado audita se o aplicativo suspeito tenta:

- detectar o nome NetBIOS do computador local. Esse nome é estabelecido na inicialização do sistema, quando o sistema o lê no registro (Regedit) [F32];
- encerrar uma chave de um registro específico [F33];
- criar uma chave de um registro específico. Caso a chave já exista no Regedit, então, ela será lida [F34];
- excluir uma chave e seus valores no Regedit [F35];
- enumerar e abrir as sub-chaves da chave de um registro aberto específico [F36].

✓ Características relacionadas a *spywares* como *keyloggers* (captura de informações do teclado visando o furto de senhas e *logins*) e *screenloggers* (filmagem da tela da vítima). O antivírus proposta audita se o arquivo analisado tenta:

- detectar, em qual parte da tela da vítima, houver uma atualização [F37];
- identificar a região de atualização da tela copiando-a para uma determinada região [F38];
- capturar filmes e vídeos AVI de câmeras da Web e outros hardwares de vídeo [F39];
- capturar informações quanto à votação eletrônica, especificamente, da empresa Optical Vote-Trakker [F40];
- copiar uma matriz de estados das teclas do teclado [F41]. Tal estratégia é típica dos *keyloggers*;
- monitorar a atividade da Internet do usuário e informações particulares [F42];
- coletar senhas bancárias *on-line* e outras informações confidenciais e enviar os dados para seu criador [F43];
- acessar um computador a partir de locais remotos, roubando senhas, transações bancárias pela Internet e dados pessoais [F44];
- criar um BHO (Objeto de Auxílio ao Navegador) o qual é executado automaticamente toda vez que o navegador web for iniciado [F45]. Cabe ressaltar que os BHOs não são impedidos por firewalls pessoais porque são identificados como parte do próprio navegador. De maneira desvirtuada, os BHOs são frequentemente usados por adware e *spyware* visando gravar entradas de teclado e do *mouse;*
- localizar senhas armazenadas em um computador [F46].

✓ Características relacionadas à Antiforense Digital as quais são técnicas de remoção, ocultação e subversão de evidências com o objetivo de reduzir as consequências dos resultados de análises forense. Caso o arquivo periciado tente:

- suspender a sua própria execução até que um determinado intervalo de tempo limite tenha decorrido [F47]. Estratégia típica dos malwares que ficam inativos até o término da quarentena dos antivírus comerciais;
- desabilitar os mecanismos de defesa da vítima, inclui-se *Firewall* e Antivírus [F48];
- desabilitar as atualizações automáticas do Windows [F49];
- detectar se o próprio arquivo está sendo analisado por um depurador do Sistema Operacional [F50];
- recuperar informações sobre o primeiro e o próximo processo encontrado em um *snapshot* do Sistema Operacional [F51]. Tal estratégia é típica de malwares que visam corromper backups e pontos de restauração do Sistema Operacional;
- ocultar um arquivo em outro. Tal estratégia é denominada, tecnicamente, de esteganografia o qual visa esconder o malware em um programa benigno no Gerenciador de Tarefas [F52];
- disfarçar o seu próprio nome no Gerenciador de Tarefas [F53];
- fazer uso de bibliotecas associadas ao *Hackers Encyclopedia 2002* [F54];
- criar um cyber-ataque do tipo ZeroAcess [F55] através de atualizações dos firmwares dos dispositivos de hardware (eg.: controlada do disco rígido).

✓ Características relacionadas à criação de GUI (*Graphical User Interface* – Interface Gráfica do Usuário) do programa suspeito. O antivírus criado audita se o arquivo suspeito tenta:

- criar uma GUI em tempo de execução [F56];
- usar o DirectX o qual permite que aplicativos multimídia desenhem gráficos 2D [F57];

- criar módulo que contém rotinas de compactação e descompactação de bitmap usadas para o Microsoft Vídeo para Windows [F58];
- criar gráficos 3D relacionadas a funções utilitárias usadas pelo OpenGL [F59];
- detectar formas através de visão computacional e processamento digital de imagem [F60];
- acessar funcionalidades visando criar e gerenciar janelas de tela e controles mais básicos, como botões e barras de rolagem, receber entrada de mouse e teclado e outras funcionalidades associadas à parte de GUI do Windows. Isso inclui coisas como barras de status, barras de progresso, barras de ferramentas e guias [F61];

✓ Características relacionadas à perícia ilícita da memória principal (RAM) do sistema local. O antivírus criado investiga se o aplicativo suspeito tenta:

- acessar informações em regiões específicas da memória principal [F62];
- ler dados de uma área de memória ocupada por um processo específico [F63];
- Gravar dados em uma área de memória em um processo específico [F64];
- Reservar, confirmar ou alterar o estado de uma região de páginas no espaço de endereço virtual de um processo [F65].

✓ Características relacionadas ao tráfego de rede. Averígua-se se o arquivo testado tenta:

- consultar servidores DNS [F66];
- enviar solicitação para um o servidor HTTP [F67];
- monitorar informações dos cabeçalhos dos pacotes de dados do computador associadas a uma solicitação HTTP [F68];
- enviar uma solicitação de eco IPv4 ICMP [F69];
- enviar uma solicitação SNMP utilizada para monitorizar equipamentos de rede local [F70];
- encerrar a conexão com a Internet [F71];
- criar uma sessão FTP ou HTTP em tempo de execução [F72];
- fragmentar uma URL em tempo de execução [F73];
- consultar um servidor para determinar a quantidade de dados disponíveis de tráfego [F74];
- identificar o estado de conexão do sistema local em relação à Internet [F74];
- inicializar o uso de um aplicativo das funções do WinINet (API do Windows visando a criação e utilização de aplicação utilizando a Internet) [F75]
- ler dados dos pacotes de rede feita a partir de requisições prévias do sistema local (comportamento típico de *sniffers*) [F76];
- sobrescrever dados em um pacote de rede do sistema local [F78];
- gerenciar sistemas de rede locais e remotos [F79];
- criar um *socket* de rede no sistema local. Em uma aplicação convencional, o servidor envia dados para o(s) cliente(s). De forma inversa, nos malwares, a vítima envia os dados (imagens, dígitos) para o servidor. Logo, os malwares podem criar sockets, no sistema local, aguardando (*listen*) que um computador mal intencionado remoto requisite uma conexão e, portanto, possa receber as informações íntimas da vítima [F80];

- Receber dados de um socket. Estratégia típica de *backdoors* quando a vítima passa a receber comandos (ordens) remotos [F81];
- Enviar dados em um socket. Estratégia típicas de *spywares* os quais, após a captura de informações íntimas, as enviam para um computador remoto mal-intencionado [F82].

✓ Características relacionadas a programas aplicativos utilitários. O antivírus criado verifica se o arquivo suspeito tenta:

- reproduzir vídeos/áudios pelo *Windows Media Player* [F83];
- alterar ícone atalho e configurações padrões de Internet exibidos na barra de endereços da barra de ferramentas do Explorer [F84];
- alterar as configurações do *Wordpad* [F85];
- alterar as configurações dos sockets, especificamente, gerenciados pelo internet Explorer [F86];
- alterar as configurações do Outlook Express e acessar a lista de contatos de e-mail da vítima [F87];
- acessar informações atreladas ao *Microsof Office* [F88];
- alterar as configurações do suíte da *Adobe Systems* [F89];
- alterar as configurações da limpeza de disco do Sistema [F90];
- alterar as configurações de jogos eletrônicos digitais nativos além dos vinculados às empresas *Tycoon* e *Electronic Arts*;
- alterar as configurações de atualizações do *Google Inc* [F91];
- usar o software Visual Basic [F92]. Tal estratégia é típica dos vírus de macro os quais são visam infectar os aplicativos que suportam linguagem de macro como os navegadores web, o *Microsoft Office* e o *Adobe Systems*.
- alterar as configurações de acesso ao Wikipédia [F93];

Através da descrição das características auditadas pelo antivírus proposto, é possível estabelecer que os malwares, em grande parte dos casos, empregam serviços lícitos nativos do sistema operacional, no entanto, de forma desvirtuada (CONRAD, *et al*., 2017). Os malwares, por exemplo, usam a webcam da vítima sem o seu consentimento. Conclui-se que é equivocado condenar um aplicativo por ele fazer uso de um determinado processo (eg. webcam). Em síntese, é incorreto condenar uma aplicação por conta de uma única característica, pois tal característica também pode ser usada por aplicações benignas. Então, o reconhecimento de malwares deve ocorrer através do cruzamento de informações e, consequentemente, a ponderação de todos os comportamentos auditados. O antivírus proposto pondera, estatisticamente, as características auditadas através do uso de *Data Science*, máquinas de aprendizado estático e inteligência artificial.

Tabela 11. Parte 1: Ferramentas, funções e bibliotecas empregadas na extração de características dos arquivos executáveis.

Código	Ferramenta
F01	Pescanner: Ferramenta de extração de características dos executáveis portáveis (PE files).

F02	Repertório de instruções da arquitetura de 32 bits do Windows. Disponível em: https://docs.microsoft.com/en-us/windows-hardware/drivers/debugger/x86-instructions.
F03	Bibliotecas nativas do Windows Npptools, Sprintsv.
F04	Função GetUserNameA da biblioteca Windows.h.
F05	Biblioteca Useren nativa do Windows.
F06	Função GetTickCount da biblioteca Windows.h.
F07	Biblioteca ShellExecute nativa do Windows.
F08	Função GetVersionEx da biblioteca Windows.h. Biblioteca powrprof.dll nativa no Windows 10.
F09	Biblioteca Hook nativa do Windows.
F10	Função GetStartupInfo da biblioteca Windows.h. Bibliotecas Psapi, Shell32, Ole32, Oleaut32, Ws2_32, msvcrt, uxtheme, Crypt32, ntdll, Rpcrt4, Odbc32, Mpr, Msvcp60, Imagehlp, netapi32, Setupapi, Iphlpapi, Dnsapi nativas do Windows.
F11	Biblioteca Shlwapi nativa do Windows.
F12	Biblioteca Msimg32 nativa do Windows.
F13	Biblioteca comdlg32 nativa do Windows.
F14	Biblioteca Slmgr nativa do Windows.
F15	Biblioteca Rpcrt4 e MSVCP60 nativas do Windows.
F16	Biblioteca Powrprof nativa do Windows.
F17	Bibliotecas GetThreadContext, OpenProcess, StartService, TerminateProcess, WinExec e Kernel32 nativas do Windows.
F18	Biblioteca CertDeleteCertificateFromStore nativa do Windows.
F19	Funções CopyFile, CeCopyFile, GetTempFileName, GetFileSize, GetFileAttributes e GetModuleHandle da biblioteca Windows.h.
F20	Funções CreateFile e DeleteFile da biblioteca Windows.h.
F21	Bibliotecas CreateProcess, ZwQueryInformationProcess , CreateToolhelp32Snapshot CreateFileMapping CreateRemoteThread CreateThread EnumProcesses ExitThread GetCommandLine GetProcAddress nativas do Windows.
F22	Função CreateDirectory da biblioteca Windows.h.
F23	Funções FindFirstFile, FindNextFile, FindFirstFile, FindFirstFileEx, e FindFirstFileTransacted da biblioteca Windows.h.
F24	Função CreateService da biblioteca Windows.h.
F25	Bibliotecas CryptEncrypt e Crypt.
F26	Bibliotecas Advapi32 e Wmiprvse nativas do Windows.
F27	Biblioteca Msacm32 nativa do Windows.
F28	Biblioteca Gdi32 e Print nativa do Windows.
F29	Processo res.exe pertencente ao USB Storage Toolbox nativo do Windows.

Tabela 12. Parte 2: Ferramentas, funções e bibliotecas empregadas na extração de características dos arquivos executáveis.

Código	Ferramenta
F30	Funções FindResource e FltRegisterFilter da biblioteca Windows.h.
F31	Funções GetDriveType e WriteFile nativas da biblioteca Windows.h.
F32	Função GetComputerName da biblioteca Windows.h.

F33	Biblioteca RegCloseKey nativa do Windows.
F34	Biblioteca RegCreateKeyA nativa do Windows.
F35	Bibliotecas RegDeleteKey e RegDeleteValue nativas do Windows.
F36	Bibliotecas RegEnumKey e RegOpenKey nativas do Windows.
F37	Função GetUpdateRect da biblioteca Windows.h.
F38	Função GetUpdateRgn da biblioteca Windows.h.
F39	Biblioteca avicap32 nativa do Windows.
F40	Biblioteca MSVBVM60 desenvolvida pela Avante International Technology.
F41	Bibliotecas SetKeyboardState, GetKeyboardState e GetKeyState nativas do Windows. Biblioteca msto32 do cavalo de Tróia Tofger-N.
F42	Biblioteca mswapi vinculada ao Trojan.Iespy-A.
F43	Biblioteca user128 vinculada ao infostealer.pport trojan.
F44	win.exe é um processo que está registrado como o cavalo de troia Agobot. aim spy plugin.dll é uma biblioteca que está registrada como o Trojan AIMVision pelo Digital Underground.
F45	Bibliotecas Htmledit.dl e toolband.dll.
F46	Ferramenta: Win32 / WEPSpy.
F47	Biblioteca Sleep nativa do Windows.
F48	Biblioteca Secur32 nativa do Windows. Biblioteca Icq.dll.
F49	Biblioteca Urlmon.dll nativa do Windows.
F50	Bibliotecas IsDebuggerPresent, OutputDebugString, CheckRemoteDebuggerPresent e UnhandledExceptionFilter nativas do Windows;
F51	Bibliotecas Process32First Process32Next.
F52	Processo lsasrv32.exe mal-intencionado. Biblioteca Hksrv.
F53	Biblioteca wininet que pode ser usado pelo trojan Troj / Zlob-AO.
F54	Biblioteca packet.dll.
F55	Biblioteca hal.dll. O HAL (Camada de Abstração de Hardware) serve como uma interface entre o hardware e o software de um sistema
F56	Função GetWindowThreadProcessId da biblioteca Windows.h. Biblioteca FindWindow nativa do Windows.
F57	Bibliotecas ddraw e Dsound.
F58	Biblioteca Jpeg1x32 desenvolvida pela Eastman Software, Inc., A Kodak Business para o Windows.
F59	Bibliotecas Gdiplus, Glu32 e Opengl32.
F60	Biblioteca Msvcp71 associado ao Microsoft® Visual Studio.NET, desenvolvido pela Imagemagick. Biblioteca Quartz.dll associada ao Photosuite SE
F61	Biblioteca comctl32 nativa do Windows.
F62	Função LockResource nativa do Windows.

Tabela 13. Parte 3: Ferramentas, funções e bibliotecas empregadas na extração de características dos arquivos executáveis.

Código	Ferramenta
F63	Função ReadProcessMemory nativa do Windows.
F64	Função WriteProcessMemory nativa do Windows.

F65	Funções VirtualAlloc, VirtualAllocEx e VirtualProtect nativos do Windows.
F66	Funções DNSQuery, Httpfilter, GetHostByAddr, GetHostByName e GetHostName nativos do Windows.
F67	Funções HttpSendRequestA e Httppostda biblioteca Windows.h.
F68	Função HttpQueryInfoA da biblioteca Windows.h.
F69	Biblioteca IcmpSendEcho.
F70	Biblioteca snmpapi.dll.
F71	Função InternetCloseHandle da biblioteca Windows.h.
F72	Funções InternetConnect FtpGetFile FtpOpenFile da biblioteca Windows.h.
F73	Biblioteca InternetCrackUrlA.
F74	Biblioteca InternetQueryDataAvailable.
F75	Biblioteca InternetGetConnectedState.
F76	Biblioteca InternetOpenA.
F77	Biblioteca InternetReadFile.
F78	Biblioteca InternetWriteFile.
F79	Biblioteca WMIPRVI: Windows Management Instrumentation.
F80	Funções listen, socket, WSASend, WSASocket e WSAStartup. Bibliotecas Wsock32 e Ws2help.
F81	Função recv nativa do Windows.h.
F82	Função send nativa do Windows.h.
F83	Biblioteca Wmvcore nativa do Windows.
F84	Bibliotecas Ur,l Comdlg32, IEHelper e Oleacc nativas do Windows.
F85	Biblioteca Oledlg nativa do Windows.
F86	Bibliotecas Mswsock e Ieloader nativas do Windows.
F87	Bibliotecas Atl, WAB32 e wab32.
F88	Bibliotecas MSVCIRT e MSO9.
F89	Processo bridge.exe e biblioteca Plugin.dll.
F90	Processo cleanmgr.exe nativo do Windows.
F91	Biblioteca goopdate.
F92	Biblioteca msvbvm50.
F93	Cygcrypt-0.dll é um tipo de arquivo DLL associado ao Wiki Web Collaboration, desenvolvido pela Springer-Verlag para o Windows.

3.3. Classificadores

Quanto ao reconhecimento de padrão de malwares, uma tarefa essencial diz respeito à atribuição de uma classe (rótulo) a cada arquivo investigado a partir de suas características. Então, com base em um conjunto de arquivos, chamado de conjunto de treinamento, é possível formular uma hipótese sobre as distintas classes atreladas ao antivírus inteligente proposto. Logo, cabe ao classificador estimar a classe de um arquivo inédito através da comparação entre as características do seu comportamento auditado e àquelas captadas durante a sua etapa de treinamento.

O objetivo do classificador é obter uma função de separação entre as classes do antivírus criado (malware, benigno). Dessa forma, ao ser apresentado a um aplicativo inédito, a função é aplicada e, então, atribui-se uma classe na qual esse aplicativo suspostamente pertence. Matematicamente, $c = f(x)$, onde $x = x_1, x_2, \ldots, x_t$ é um vetor

características extraídas do arquivo investigado, t corresponde às 630 características auditadas, c é a classe (rótulo), por fim, f é a função de mapeamento do classificador.

O trabalho proposto emprega redes neurais artificiais como classificadores. Especificamente, são empregadas as redes neurais do tipo MLP (*Multilayer Perceptron* - Perceptron com Múltiplas Camadas) dotadas de retropropagação. Uma rede MLP é uma generalização da rede Perceptron simples pela adição de, ao menos, uma camada intermediária (HAGAN, *et al*., 1996). Convencionalmente, as redes MLP apresentam 3 (três) camadas: uma camada de entrada, ao menos uma camada intermediária e uma camada de saída. Na camada de entrada, cada neurônio de entrada representa uma variável considerada como entrada para o problema. No antivírus inteligente criado, há 630 neurónios e dizem respeito a comportamentos maliciosos descritos na subseção 3.2. Na camada escondida, há o encargo pela não linearidade da rede neural. Por fim, na camada de saída, há o reconhecimento do padrão quanto à resposta da rede e representa a variável desejada. No antivírus inteligente criado, a camada de saída possui dois neurônios correspondentes aos aplicativos benignos e *malwares*.

As redes MLP, baseadas em retropropagação, utilizam basicamente dois passos. No primeiro passo, há a propagação de dados (impulso sináptico) da camada de entrada para a camada de saída. Nesse passo, há o cálculo do sinal de saída e o erro (HAGAN, *et al*., 1996). O erro é a diferença entre o sinal de saída obtida e a saída desejada. Após isso, no segundo passo, há a propagação de dados partindo da camada de saída em direção à camada de entrada. Esse passo é conhecido com retropropagação e visa ajustar os pesos referentes às ligações sinápticas entre os neurônios. Cabe ressaltar que os pesos são determinados, inicialmente, de forma aleatória.

Com o objetivo de otimizar a precisão do antivírus criado, são averiguadas onze diferentes funções de aprendizado f baseadas em classificadores MLP dotados de retropropagação. A seguir, são detalhadas todas as onze diferentes formas de implementações das redes MLPs empregadas pelo trabalho proposto.

1. Lotes de peso e limite bias: emprega atualizações em lotes, que são atualizados ao final de cada época (HAGAN, *et al*., 1996).
2. Gradiente *Powell-Beale*: emprega o método de retropropagação do gradiente conjugado baseado na técnica *Powell-Beale* (POWELL, 1977).
3. Retropropagação Fletcher-Powell: utiliza o método de retropropagação do gradiente conjugado baseado na técnica Fletcher-Reeves (NOTAY, 2000).
4. Retropropagação Polak-Ribiere: aplica o método de retropropagação do gradiente conjugado inspirado na técnica Polak-Ribiere (SCALES, 1985).
5. Retropropagação decrescente: emprega o método de retropropagação baseado em um gradiente descendente (HAGAN, *et al*., 1996).
6. Retropropagação com momentos: emprego o método de retropropagação baseado em um gradiente descendente com momentos. Esse classificador permite que a rede neural não apenas se baseie em informações locais mas também tenha a capacidade de ignorar ruídos do gradiente. O objetivo é que a rede neural não se atenha a um mínimo local (HAGAN, *et al*., 1996).

7. Retropropagação com taxa adaptativa: implementa uma taxa de aprendizagem adaptativa a qual se guia baseada nas informações do gradiente. Uma taxa de aprendizagem fixa pode apresentar problemas durante o treinamento. Um taxa de aprendizagem fixa muito alta, o classificador pode oscilar e se tornar instável. Por outro lado, caso a taxa de aprendizagem fixa for muito baixa, o tempo de treinamento pode se tornar bastante elevado (HAGAN, *et al*., 1996).
8. Retropropagação combinada: é a junção da retropropagação com taxa adaptativa com a retropropagação baseada em momentos. Isso quer dizer, essa rede neural não se baseia apenas em informações locais como também possui um taxa de aprendizagem adaptativa (HAGAN, *et al*., 1996).
9. Retropropagação secante em um passo: implementa a retropropagação do gradiente conjugado baseado na técnica da secante em um único passo (BATTITI, 1992).
10. Retropropagação resiliente (Rprop): emprega o método de retropropagação do gradiente conjugado baseado no algoritmo Rprop (RIEDMILLER, et al., 1993).
11. Retropropagação gradiente escalonado: emprego o método de retropropagação do gradiente conjugado escalonado (MOLLER, 1993).

Nas redes neurais MLP, condições iniciais convenientes podem fazer com que a rede convirja em poucas épocas. Por outro lado, condições iniciais inadequadas podem demandar uma grande quantidade de épocas até a rede adquirir capacidade de generalização e, portanto o treinamento ser encerrado. Então, um cuidado metodológico, adotado pelo trabalho proposto, é averiguar a influência das condições iniciais nos resultados alcançados dos classificadores. Logo, são utilizados 30 (trinta) diferentes pesos iniciais aleatórios referentes às ligações sinápticas entre os neurônios. As variações são aleatórias com a semente do gerador aleatório entre 1 e 30 de maneira incremental.

A quantidade de pesos iniciais foi estabelecida como 30 (trinta) sementes de gerador aleatório. A motivação é que 30 (trinta) iterações constituem uma quantidade estatisticamente significativa de modo a se determinar o grau de dispersão das amostras. Então, para cada rede neural (função de aprendizado), são avaliadas 30 (trinta) amostras assim como ocorrem em diversas áreas como, por exemplo, em Engenharia Biomédica (LIMA, *et al*., 2016). Na Tabela dos resultados, no capítulo seguinte, a dispersão é representada através do desvio padrão das 30 (trinta) iterações de cada função de aprendizado avaliada. Então, quanto menor o desvio padrão implica que a rede neural avaliada não sofre mudanças abruptas em função dos pesos iniciais aleatórios referentes às ligações sinápticas entre os neurônios. De modo oposto, quanto maior for o desvio padrão significa que a rede neural sobre brusca alteração, em sua precisão, em função dos pesos iniciais das ligações sinápticas.

O trabalho proposto faz uso de três arquiteturas para cada MLP. A primeira arquitetura utiliza uma única camada escondida, com 100 neurônios. A segunda arquitetura utiliza duas camadas escondidas, com 100 neurônios cada. Os resultados da segunda abordagem são simbolizados com um asterisco. Por fim, a terceira arquitetura emprega uma camada escondida, no entanto, são utilizados 500 neurônios. A terceira abordagem é simbolizada através de dois asteriscos na tabela que exibirá os resultados alcançados. A hipótese é verificar se o aumento da quantidade de neurônios e, consequentemente, o aumento de cálculos numéricos da rede neural é capaz de prover

melhores resultados. Para cada função de aprendizado, há 90 (3 arquiteturas * 30 pesos iniciais) iterações. São exploradas 3 (três) tipos de arquiteturas e 30 (trinta) diferentes pesos iniciais, referentes às ligações sinápticas entre os neurônios, para cada arquitetura.

Quanto aos dados de entrada, 70% dos dados são destinados ao treinamento, 15% são reservados à validação cruzada, por fim 15% são utilizados na fase de testes. O número máximo de épocas, para treinamento, é 1000 (um mil). A validação cruzada ocorre após cada época com o objetivo da rede neural não perder a sua capacidade de generalização. Há uma tolerância de 5 (cinco) falhas para a validação cruzada. O tempo de treinamento das MLPs engloba, obviamente, o tempo gasto no treinamento juntamente com o tempo destinado à validação cruzada.

4. Resultados

A Tabela 14 exibe a classificação dos resultados usando as redes MLP baseadas em retropropagação. Os melhores casos são grifados em negrito. O primeiro e segundo critérios são a média aritmética e o desvio padrão, respectivamente. São utilizadas onze diferentes funções de aprendizado. Para cada função, são estudadas três tipos distintos de arquitetura, descritas na seção 3.2.1. Quanto ao tempo de treinamento, os classificadores apresentam tempos elevados assim como os desvios padrões. Esse fato indica que as MLPs sofrem variações abruptas, em relação ao tempo de treinamento, a depender dos pesos iniciais referentes às ligações sinápticas entre os neurônios. Conforme estabelecido anteriormente, “”, “*”, “**” correspondem às arquiteturas com 1 (uma) camada escondida com 100 neurônios, 2 (duas) camadas escondidas cada qual com 100 neurônios e 1 (uma) camada escondida com 500 neurônios, respectivamente.

No tocante à precisão, a mínima taxa de acerto foi de 48,87% para a rede que emprega o Rprop (Retroprogação Resiliente). Essa abordagem possui uma arquitetura com duas camadas escondidas com 100 neurônios cada uma. Enquanto, a máxima precisão foi de 98,13% para a rede inspirada em retropropagação de Gradiente Escalonado. A arquitetura, dessa rede, contém uma camada escondida com 100 neurônios. Note também que o desvio padrão é de 0,68, um valor relativamente baixo. Isso corrobora o excelente desempenho da função de aprendizado do Gradiente Escalonado, independente das condições iniciais.

Quanto ao tempo de treinamento, os classificadores baseados em retropropagação apresentam desvios padrões altos. Isso indica que as redes MLPs sofrem variações abruptas, em relação a tempo de aprendizado, a depender do conjunto de pesos iniciais. Observa-se que, em regra geral, ao expandir a quantidade de neurônios na camada escondida, também cresce o tempo de treinamento devido a um maior volume de cálculos da rede neural.

Quanto às arquiteturas das redes, aumentar a quantidade de camadas escondidas ou aumentar a sua quantidade de neurônios, não implica em um melhor desempenho de maneira sistemática. Em várias situações, aumentar a quantidade de neurônios na camada escondida provocou uma degradação no desempenho acompanhada de um maior tempo destinado ao treinamento devido ao grande volume de cálculos. Logo, a exploração de diferentes tipos de arquiteturas de redes neurais foi uma decisão salutar visto que não houve uma arquitetura ótima independente da variação da função de aprendizado.

Tabela 14. Resultados das redes baseadas em retropropagação para classificação em duas classes de executáveis: benignos e malwares.

	Acerto treino (%)	*Acerto teste (%)*	*Tempo treino (seg.)*	*Tempo teste (seg.)*
Lotes de peso e limite bias	54,12 ± 14,30	54,11± 14,11	27,90±79,20	0,06±0,04
Lotes de peso e limite bias*	56,27 ± 15,38	56,21± 15,21	26,60±84,70	0,06±0,04
Lotes de peso e limite bias**	53,81 ± 11,43	53,79± 11,35	40,16±199,11	0,11±0,4
Gradiente Powell-Beale	96,95 ± 4,19	96,31 ± 4,28	41,44±24,33	**0,05±0,03**
Gradiente Powell-Beale*	96,75 ± 3,60	96,21 ± 3,66	41,65±23,43	0,07±0,05
Gradiente Powell-Beale**	96,70 ± 4,09	96,25 ± 3,92	76,19±40,99	0,10±0,04
Retropropagação Fletcher-Powell	97,41 ± 4,64	96,55 ± 4,60	61,79±32,72	0,06±0,03
Retropropagação Fletcher-Powell*	98,32 ± 2,95	97,60 ± 2,99	85,20±43,79	0,05±0,04
Retropropagação Fletcher-Powell**	98,33 ± 3,40	97,54 ± 3,42	165,77±75,37	0,08±0,04
Retropropagação Polak-Ribiere	96,96 ± 34,6	96,39 ± 3,47	49,81±25,73	**0,05±0,03**
Retropropagação Polak-Ribiere*	97,15 ± 3,00	96,69 ± 3,01	62,92±30,45	0,06±0,05
Retropropagação Polak-Ribiere**	96,96 ± 2,11	96,54 ± 1,96	112,39±65,00	0,10±0,06
Retropropagação decrescente	83,11 ± 14,64	82,94± 14,46	43,09±55,64	0,06±0,04
Retropropagação decrescente*	77,40 ± 16,13	76,63± 16,23	61,97±89,61	0,06±0,04
Retropropagação decrescente **	52,86 ± 13,28	52,77± 12,94	19,23±82,56	0,10±0,04
Retropropagação com momentos	55,29 ± 14,34	55,17± 14,22	13,98±38,29	0,05±0,05
Retropropagação com momentos*	51,63 ± 10,90	51,67± 10,87	8,82±37,48	0,05±0,05
Retropropagação com momentos**	50,16 ± 4,72	50,28 ± 4,80	1,98±0,40	0,09±0,04
Retropropagação com taxa adaptativa	81,18 ± 07,42	80,68± 7,29	6,74±2,54	0,05±0,04
Retropropagação com taxa adaptativa*	85,63 ± 3,59	84,94± 3,24	12,44±2,66	0,07±0,04
Retropropagação com taxa adaptativa**	78,17 ± 11,40	77,96± 11,29	35,53±21,23	0,09±0,04
Retropropagação combinado	88,92±8,28	87,96±8,11	14.00±3,56	0,06±0,04
Retropropagação combinado*	89,09±6,27	88,11±6,18	19,46±7,12	0,06±0,04
Retropropagação combinado**	86,37 ± 12,56	85,69± 12,16	51,85±20,89	0,09±0,03
Retropropagação secante em um passo	98,60±1,17	98,00± 0,92	70,19±25,57	0,06±0,04
Retropropagação secante em um passo*	98,36± 1,25	97,76± 0,99	92,30±33,48	0,05±0,04
Retropropagação secante em um passo**	97,47 ± 1,50	97,04 ± 1,27	168,50±67,58	0,10±0,04
Retropropagação resiliente (Rprop)	50,67±4,16	50,70± 3,92	**0,68±0,11**	0,07±0,04
Retropropagação resiliente (Rprop)*	48,82± 7,03	48,87± 6,83	1,01±0,17	0,06±0,04
Retropropagação resiliente (Rprop)**	51,82 ± 4,61	51,57± 4,38	1,47±0,16	0,10±0,04
Retropropagação gradiente escalonado	**98,82± 0,95**	**98,13±0,68**	30,92±6,69	0,06±0,04
Retropropagação gradiente escalonado*	98,57± 1,06	98,05± 0,81	40,82±11,40	0,06±0,04
Retropropagação gradiente escalonado**	97,59± 1,51	97,17± 1,24	65,00±26,83	0,09±0,05

Então, a inteligência artificial se torna uma boa alternativa para as fabricantes dos antivírus comerciais. A inteligência artificial, durante sua etapa de aprendizado, é capaz de analisar milhares de malwares e extrair características deles. Especificamente, no antivírus inteligente proposto, são extraídas 630 características referentes ao algoritmo do arquivo suspeito. Então, após o aprendizado, caso um novo arquivo, ainda não catalogado, fosse investigado, haveria a possibilidade dele ser corretamente classificado, como *malware*. O correto reconhecimento de padrão se dá através da comparação entre as características do seu algoritmo e as aquelas captadas durante o processo de aprendizado da inteligência artificial.

Um antivírus, ao empregar inteligência artificial, consegue automatizar a análise de centenas de características de arquivos suspeitos em larga escala e em tempo real. Logo, não seria mais necessário aguardar que algum cliente fosse infectado e, em

sequência denunciasse um comportamento anômalo de seu dispositivo, para, então, tomar-se providências quanto à detecção de um novo *malware*. Cabe salientar que as malfeitorias dos malwares podem ser irreversíveis e irrecuperáveis. Logo, um antivírus deve detectar as pragas virtuais de forma preventiva ao invés de reativa. A inteligência artificial possibilita a detecção preventiva da praga virtual antes mesmo dela ser executada pela cliente. O antivírus inteligente criado alcança um desempenho médio de 98,13% na distinção entre executáveis benignos e *malwares*, acompanhado de um tempo de resposta médio de apenas 0,06 segundos.

5. Conclusão

Sistematicamente, ao longo da década, a produção de malwares é crescente (CERT.BR, 2016). Então, é importante que os antivírus provenham serviços robustos, em larga escala e em tempo real, com o objetivo de coletar amostras maliciosas recém-criadas visando proteger seus respectivos clientes. Conclui-se que a escolha de um antivírus adequado exerce papel significante na proteção contra invasões cibernéticas, visto que a detecção de *malwares* variou entre 0% e 99,11%, a depender do antivírus comercial investigado, conforme detalhado no capítulo 2. O trabalho proposto investigou 86 antivírus comerciais. Em média, eles foram capazes de detectar 54,84% das pragas virtuais avaliadas. Com aspecto desfavorável, os antivírus, em média, atestaram falsos negativos e foram omissos em 14,34% e 30,82% dos casos, respectivamente.

O trabalho proposto empregou o VirusTotal como plataforma para submissão automatizada dos *malwares* aos antivírus comerciais. No VirusTotal, não há a possibilidade de escolha da versão gratuita dos antivírus. Logo, como limitação do estudo proposto, não foi possível averiguar a diferença dos serviços providos pelas versões pagas e as gratuitas de um mesmo antivírus. A suposição é que as versões gratuitas disponibilizam serviços significativamente inferiores às versões pagas.

Cerca de 17% dos antivírus avaliados não foram capazes de diagnosticar qualquer uma das amostras maliciosas. Nota-se, a limitação dos antivírus comerciais quanto à robustez de serviços em larga escala mesmo em suas versões completas (pagas). Cabe salientar que os malwares, pertencentes à base criada REWEMA, são de domínio público, largamente empregados em atividades maliciosas e com as suas atuações amplamente divulgadas em fóruns, blogs e demais conteúdos online. Mesmo assim, 17% dos antivírus comerciais avaliados não tinham qualquer conhecimento sobre as suas existências.

O trabalho proposto desenvolve um antivírus, dotado de inteligência artificial, visando a classificação quanto a executáveis de arquiteturas de 32 bits entre benignos e *malwares*. A extração de características passa pelo processo de *disassembling* visando reverter o arquivo binário em seu código fonte. A classificação é baseada em redes neurais baseadas em retropropagação. As condições iniciais e arquiteturas são alternadas com o objetivo de maximizar a precisão dos classificadores.

Quanto à classificação, a função de aprendizado dotada de retropropagação de Gradiente Escalonado apresenta a maior precisão dentre todos os classificadores averiguados, com uma precisão média de 98,13%. Essa abordagem possui uma arquitetura com única camada escondida de 100 neurônios. O classificador de mais baixa precisão foi de 48,87%, referente à função de aprendizado Rprop (Retroprogação Resiliente). Esse pior cenário ocorre quando o Rprop é formado por uma arquitetura com duas camadas escondidas com 100 neurônios. Conclui-se que a melhor abordagem é

quase 100,00% superior ao pior cenário obtido. Logo, a escolha de uma adequada função de aprendizado, composta por uma correta arquitetura, é essencial para maximizar a precisão quanto à identificação de *malwares*.

A grande contribuição do trabalho proposto, em relação ao estado-da-arte, é suprir as limitações e imprecisões dos modelos empíricos e heurísticos comumente empregados na detecção de pragas virtuais (PRADO, *et al*., 2016). Então, ao invés da intuição do analista, o peso ponderado referente a um comportamento suspeito é determinado através de redes neurais artificiais. O trabalho proposto constata que a inteligência artificial pode contribuir bastante para o avanço da segurança da informação em dispositivos digitais. Espera-se que o antivírus inteligente proposto atue de forma preventiva e impeça que os malwares burlem os mecanismos de defesa da vítima, inclui-se firewall e plugins de segurança.

Como limitação, o antivírus inteligente criado somente é capaz de detectar malwares, especificamente, para arquitetura de 32 bits do Sistema Operacional Windows. A explicação é que cada Sistema Operacional tem seu próprio repertório de instruções, bibliotecas e APIs. Então, por exemplo, os repertórios de instruções do Windows e do Linux são diferentes. Logo, a extração de características para aplicativos Windows, conforme detalhada na seção 3.2, não serve para Linux ou para Android. Conclui-se que o antivírus inteligente proposto não teria validade mediante malwares dos sistemas Linux e Android.

Então, a meta futura diz respeito à criação de novos antivírus, baseados em inteligência artificial, no sentido de suas aplicações a outros tipos de sistemas operacionais além do Windows. A intenção é estender a metodologia proposta a arquivos executáveis do sistema Android visto que s*martphones* e *tablets* móveis estão gradativamente se tornando indispensáveis na vida diária. O *Android*, por ser um sistema operacional relativamente recente, possibilita que incontáveis *malwares* se escondam em uma grande quantidade de aplicações legítimas, o que ameaça de forma grave a segurança do sistema e afeta diretamente o usuário.

Referências

ADKINS, F.; JONES, L.; MARTIN, C.; UPCHURCH, J. Heuristic Malware Detection Via Basic Block Comparison. 8th International Conference on In Malicious and Unwanted Software: "The Americas"(MALWARE), páginas 11–18, 2013.

ALAZAB, M. Profiling and Classifying the Behavior of Malicious Codes Journal of Systems and Software. Volume 100, Páginas 91–102, 2015.

BATTITI, R. First and second order methods for learning: Between steepest descent and Newton's method. Neural Computation, volume. 4, número 2, pág. 141–166 , 1992.

CERT.BR. Incidentes Reportados ao CERT.br. Disponível em: http://www.cert.br/stats/incidentes/2014-jan-dec/analise.html. Acesso em outubro de 2016, 2016.

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

FAN, Y.; YE, Y.; CHEN, L. Malicious Sequential Pattern Mining for Automatic Malware Detection. Expert Systems with Applications. Volume 52, número 15, páginas 16–25, 2016.

FILHO, D. S. F.; AFONSO, V. M.; MARTINS, V. F.; GRÉGIO, A. R. A.; GEUS, P. L.; JINO, M.; SANTOS, R. D. C. Técnicas para Análise Dinâmica de Malware. Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais. ISBN: 978-85-7669-259-1, 2011.

HAGAN, M. T.; H.B., D.; M.H., B. Neural Network Design. Boston, MA: PWS Publishing, 1996.

HENKE, M.; SANTOS, C.; NUNAN, E.; FEITOSA, E.; SANTOS, E.; SOUTO, E. Aprendizagem de Máquina para Segurança de Computadores: Métodos e Aplicações. Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais. ISBN: 978-85-7669-259-1, 2011.

INTEL. McAfee Labs: Relatório sobre ameaças. Disponível em: https://secure.mcafee.com/br/resources/reports/rp-quarterly-threats-mar-2017.pdf?cid=BHP-075, 2017.

KHODAMORADI, P.; FAZLALI, M.; MARDUKHI, F.; AND NOSRATI, M. Heuristic Metamorphic Malware Detection based on Statistics of Assembly Instructions using Classification Algorithms. 18th CSI International Symposium on Computer Architecture and Digital Systems (CADS), páginas 1-6, 2015.

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. Volume 134, página 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 2018 , 2014.

MOLLER, F. M. A scaled conjugate gradient algorithm for fast supervised learning Neural Networks. Volume 6, pág. 525–533, 1993.

NOTAY, Y. Flexible Conjugate Gradients. SIAM Journal on Scientific Computing 22 (4): 1444. doi:10.1137/S1064827599362314 , 2000.

POWELL, M. J. D. Restart procedures for the conjugate gradient method. Mathematical Programming. Volume 12, pág. 241–254, 1977.

PRADO, N.; PENTEADO, U.; GRÉGIO, A. Metodologia de Detecção de Malware por Heurísticas Comportamentais. Simpósio Brasileiro em Segurança da Informação e de Sistemas. ISSN: 2176-0063, 2016.

REWEMA. REWEMA (Retrieval of 32-bit Windows Architecture Executables Applied to Malware Analysis – Redistribuição de Executáveis do Windows em Arquiteturas de 32-bits Aplicados a Análise de Malware) Disponível em: https://github.com/rewema/rewema. Acesso em janeiro de 2019.

RIEDMILLER, M.; BRAUN, H. A direct adaptive method for faster backpropagation learning: the RPROP algorithm. Proceedings of the IEEE International Conference on Neural Networks (ICNN). San Francisco, pág. 586–591, 1993.

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 2017, 2018.

SCALES, L. E. Introduction to Non-Linear Optimization. New York, Springer-Verlag, 1985.