UNIVERSIDADE DO SAGRADO CORAÇÃO

ANDRÉ LUÍS SANCHES BENINI

ANÁLISE DAS TÉCNICAS ESTÁTICA E DINÂMICA PARA DETECÇÃO DE MALWARE

BAURU
2017

ANDRÉ LUÍS SANCHES BENINI

ANÁLISE DAS TÉCNICAS ESTÁTICA E DINÂMICA PARA DETECÇÃO DE MALWARE

Trabalho de Conclusão de Curso apresentado ao Centro de Ciências Exatas e Sociais Aplicadas da Universidade do Sagrado Coração, como parte dos requisitos para obtenção do título de Bacharel em Ciência da Computação, sob orientação do Prof. Me. Henrique Pachioni Martins.

BAURU
2017

ANDRÉ LUÍS SANCHES BENINI

ANÁLISE DAS TÉCNICAS ESTÁTICA E DINÂMICA PARA DETECÇÃO DE MALWARE

Trabalho de Conclusão de Curso apresentado ao Centro de Ciências Exatas e Sociais Aplicadas da Universidade do Sagrado Coração, como parte dos requisitos para obtenção do título de Bacharel em Ciência da Computação, sob orientação do Prof. Me. Henrique Pachioni Martins.

Bauru, 22 de novembro de 2017.

Banca examinadora:

_____
Prof. Me. Henrique Pachioni Martins
Universidade do Sagrado Coração

_____
Prof. Me. Renan Caldeira Menechelli
Universidade do Sagrado Coração

_____
Prof. Dr. Elvio Gilberto da Silva
Universidade do Sagrado Coração

Benini, André Luís Sanches

B4674a

Análise das técnicas estática e dinâmica para detecção de malware / André Luís Sanches Benini. -- 2017.

63f. : il.

Orientador: Prof. Me. Henrique Pachioni Martins.

Trabalho de Conclusão de Curso (Graduação em Ciência da Computação) – Universidade do Sagrado Coração – Bauru – SP.

1. Malware. 2. Segurança da Informação. 3. Análise estática. 4. Análise dinâmica. I. Martins, Henrique Pachioni. II. Título.

AGRADECIMENTOS

Primeiramente, agradeço imensamente aos meus pais, Carlos Alberto Benini e Arlete Toledo Sanches Benini, por sempre me apoiarem e me darem as melhores condições possíveis para que eu pudesse ter chegado a este momento, por toda educação e ensinamentos me dado ao longo desta vida. Agradeço também a minha irmã, Maria Beatriz, por sempre ter sido uma parceira e estado junto em todos os momentos.

Agradeço também, a todos os meus professores ao longo da vida, certamente eu não chegaria aqui sem nenhum conhecimento transmitido por eles. Agradeço muito também, a todos os professores durante esses 4 anos, que foram fundamentais para minha formação em bacharel em ciência da computação, em especial, meu orientador Prof. Me. Henrique Pachioni Martins.

Não poderia esquecer de agradecer também a todos meus colegas de turma, principalmente a Felipe, Rodolfo, Moacir, José Augusto e Elyhan, por sempre estarem ao meu lado e ajudando uns aos outros, que com certeza os levarei ao logo de toda minha vida.

“Seja você quem for, seja qual for a posição social que você tenha na vida, a mais alta ou a mais baixa, tenha sempre como meta muita força, muita determinação e sempre faça tudo com muito amor e com muita fé em Deus, que um dia você chega lá. De alguma maneira você chega lá.”
(Ayrton Senna).

RESUMO

Na área da segurança da informação, muitas são as ameaças existentes, sendo a maior delas, em termos de quantidade e periculosidade, o malware. Este termo utilizado para códigos maliciosos de uma maneira geral, cujo os objetivos são destruir, alterar, espionar, clonar, sobrecarregar, entre outras funções. Por isso, torna-se fundamental a detecção do malware, para não houver prejuízos. Desta forma, sua detecção é realizada através de análises, uma estática, em que não se executa o objeto que está sendo analisado, outra dinâmica, que realiza durante sua execução. Com este motivo, este trabalho se propôs a realizar e demonstrar como estas análises funcionam e são importantes para a segurança da informação, identificando e demonstrando seus processos, sempre buscando adquirir e passar informações e conhecimentos, uma outra visão, buscando o crescimento e a contribuição com a segurança da informação. Assim, a metodologia aplicada, foi devido ao levantamento bibliográfico, realizando análise estática e dinâmica sobre um malware em um ambiente virtual seguro criado, extraindo, compreendendo, os resultados conseguidos através desta pesquisa e análises. Com isso observou todo processo de análise é complexo, que requisita paciência em uma análise de malware, sendo observado que, método o estático acaba sendo um pouco menos profundo que o dinâmico, porém, para uma análise completa, uma junção dos dois para se entender melhor o funcionamento, ameaças, e como combater o malware.

Palavras-chave: Malware. Segurança da informação. Análise estática. Análise dinâmica.

ABSTRACT

In the area of information security, many threats exist, being the biggest of which, in terms of quantity and danger, the malware. This term used for malicious code in general, which the objectives is, destroy, alter, spy, overload, among other functions. Therefore, it is essential to detect the malware, to avoid losses. In this way, the detection performed through analysis, a static, which it is not executed the object of analysis, and another dynamic, which performs during of its execution. For this reason, this research intends to perform and demonstrate how these analyzes work and are so important for security information, identifying and demonstrating their processes, always seeking to acquire and pass information and knowledge, another vision, seeking growth and contribution to security information. Thereby applied methodology was due to the bibliography survey, performing static and dynamic analysis in a malware in a secure virtual environment created, extracting, understanding, discussing, the results achieved through this research and analyzes. This way, observed all the process is complex, which requires patience in a malware analysis, being observed that, method the static ends up being a bit less deep than the dynamic, however, for a complete analysis, a combination of the two to better understanding running, threats, and how to combat malware.

Keywords: Malware. Information security. Static Analysis. Dynamic Analysis.

TABELA DE ILUSTRAÇÕES

Figura 1 - Phishing através de e-mail falso do Banco do Brasil.....19
Figura 2 - Phishing através de um site falso do Facebook.....19
Figura 3 - Usuário normal com acesso negado devido à sobrecarga.....21
Figura 4 - Estatística que mostra a quantidade de malware desenvolvido para cara tipo.....22
Figura 5 - Exemplo de relatório de um Keylogger.....24
Figura 6 – Pop-ups de propaganda exibidos por um adware. 25
Figura 7 – Exemplo de Ransomware. 27
Figura 8 – Exemplo de um dos primeiros ransomware russos.....27
Figura 9 – Atividade do rasomware com uso de trojans.....28
Figura 10 – Ransomware distribuído geograficamente.....28
Figura 11 – Ransomware WannaCry.....29
Figura 12 – Fluxograma do desenvolvimento das políticas de segurança. 34
Figura 13 – Uso de máquina virtual para criação de diversos sistemas operacionais.....37
Figura 14 – Infraestrutura da máquina virtual windows 7. 41
Figura 15 – Diagrama da metodologia.....43
Figura 16 – Hash do arquivo encontrada. 45
Figura 17 – Hash encontrada do arquivo apontando malware. 46
Figura 18 – Taxa de antivírus que o detectam. 46
Figura 19 – Arquivo detectado por uns e por outros não. 47
Figura 20 – Pacote do C++ identificado. 48
Figura 21 – Arquivo descomprimido. 48
Figura 22 – Código ilegível a ser desconsiderado. 49
Figura 23 – Arquivos DLL encontrados no código. 50
Figura 24 – DLLs importadas do arquivo. 51
Figura 25 – Funções da DLL. 51
Figura 26 – Resultados análise estática.....52
Figura 27 – Primeira captura. 53
Figura 28 – Relatório do Regshot. 54
Figura 29 – DLLs modificadas. 54
Figura 30 – Malware conectando site. 55

Figura 31 – Malware buscando DLLs. 55
Figura 32 – Malware processando DLL. 56
Figura 33 – Resultados análise dinâmica 57
Figura 34 – Comparativo das análises 57

TABELA DE ABREVIATURAS E SIGLAS

DDos	Distributed Denial of Service
DLL	Dynamic Link Library
DoS	Denial of Service
KB	Kilobyte
MS-DOS	Microsoft Disk Operating System
RAM	Random Access Memory
USB	Universal Serial Bus

SUMÁRIO

1 INTRODUÇÃO 14

2 OBJETIVOS 15

2.1 OBJETIVO GERAL 15

2.2 OBJETIVOS ESPECÍFICOS 15

3 SEGURANÇA DA INFORMAÇÃO 16

3.1 PILARES DA SEGURANÇA DA INFORMAÇÃO 16

3.2 VULNERABILIDADE 16

3.3 AMEAÇAS 17

4 TÉCNICAS DE ATAQUES 18

4.1 ATAQUES PASSIVOS 18

4.2 PHISHING 18

4.3 DENIAL OF SERVICE 20

4.4 MALWARE 21

4.4.1 História do Malware 22

4.4.2 Tipos de Malware 23

4.4.2.1 Vírus de Computador 23

4.4.2.2 Worms 23

4.4.2.3 Cavalo de Troia 23

4.4.2.4 Keylogger 24

4.4.2.5 Spyware 24

4.4.2.6 Adware 24

4.4.2.7 Backdoor 26

4.4.2.8 Exploits 26

4.4.2.9 Rootkit 26

4.4.2.10 Ransomware 26

4.4.2.10.1 Ransomware de ataque global 29

4.4.3 Evolução 29

4.4.3.1 Malware Cifrados 29
4.4.3.2 Malware Oligomórfica 30
4.4.3.3 Malware Polimórfico 30
4.4.3.4 Malware metamórfico 30
4.4.4 Análise e Detecção 31
4.4.4.1 Engenharia Reversa 31
4.4.4.2 Análise Estática 31
4.4.4.3 Análise Dinâmica 32
4.5 PREVENÇÃO 33
4.5.1 Política de Segurança 33
4.5.2 Antivírus 35
4.5.3 Antispyware 35
4.5.4 Firewall 35
5 MÁQUINAS VIRTUAIS 37
5.1 WINDOWS 38
5.1.1 Windows 7 38
5.2 ANALISE DE MALWARE EM MÁQUINA VIRTUAL 38
6 TRABALHOS CORRELATOS 39
7 METODOLOGIA 41
8 RESULTADOS E DISCUSSÕES 45
8.1 ANALISE ESTÁTICA 45
8.1.1 Hash 45
8.1.2 Verficação em banco de dados 46
8.1.3 Packers 47
8.1.4 Strings 49
8.1.5 PE Headers 50
8.1.6 Discussão dos resultados 52
8.2 ANALISE DINÂMICA 53

8.2.1 Captura das informações do sistema inicial 53

8.2.2 Execução e análise do malware 54

8.3 COMPARATIVO 57

9 OBSERVAÇÕES FINAIS 58

REFERENCIAS 59

1 INTRODUÇÃO

Este trabalho, tem como foco a área de segurança da informação, de forma mais precisa, em *malware*, que são *softwares* maliciosos, construídos para roubo de informações e dados, derrubar ou prejudicar sistemas, invasão de privacidade entre outros. A cada dia, milhares de *malware* são desenvolvidos, cada vez mais perigosos e mais difíceis de serem detectados. Para sua detecção, existem dois métodos: estático e dinâmico. De forma sucinta, a diferença entre elas é, enquanto a estática analisa o *malware* sem a sua execução, a dinâmica o faz executando.

Neste trabalho foi analisado as formas de detecção de um *malware*, seus métodos de processo e análise, categorizar suas diferenças, através de, implem

entações em uma máquina virtual. Segundo Sikorski e Honig (2012, p. 28) “A análise de *malware* é a arte de dissecar o malware para entender como funciona, como identificá-lo, e como derrotá-lo ou eliminá-lo”. Sua análise é importante processo na segurança da informação, pois é nela em que é obtido informações de onde ele surgiu, qual seu objetivo de ataque, saber o dano causado por ele, e de que forma se prevenir. Além disso, com sua identificação, se torna mais fácil e rápido encontrar variações e mutações daquele *malware*, facilitando uma detecção futura.

A importância desta pesquisa, foi estudar e trazer conhecimentos sobre os métodos de detecção, demonstrando seu funcionamento e sua prática, através de realização de testes e analisado *malware*, para que se possa obter e demonstrar os resultados obtidos através destas análises.

2 OBJETIVOS

Neste tópico, se apresentam dos objetivos que foram definidos para esta pesquisa, divididos e objetivo geral e específicos.

2.1 OBJETIVO GERAL

Analisar diferentes técnicas de detecção de *malware*, compreender seu funcionamento e a forma que são executadas, além de fazer um comparativo entre elas.

2.2 OBJETIVOS ESPECÍFICOS

a) Realizar um levantamento bibliográfico sobre as técnicas de detecção.
b) Estudar a teoria e prática de seu funcionamento.
c) Pesquisar sobre *malware* e outras ameaças, e como são desenvolvidos e funcionam.
d) Implementar um ambiente virtual e instalação de softwares necessários.
e) Obter *malware* para a realização de testes, e realizar seus métodos de detecção.

3 SEGURANÇA DA INFORMAÇÃO

A segurança da informação, é a área da computação que trata sobre todos os métodos, técnicas, normas e regras sobre segurança de um sistema, de dados, de arquivos, ou tudo aquilo que é considerado importante e necessário, para uma empresa, ou um usuário comum, com o intuito de protege-lo, tanto virtualmente, mas como também fisicamente.

Para Alexandria (2009), a segurança da informação, significa a proteção da informação do acesso desautorizado, de uma modificação ou destruição.

3.1 PILARES DA SEGURANÇA DA INFORMAÇÃO

De acordo com Fernandes (2010), consideram-se os pilares de segurança da informação, a confiabilidade, a integridade, a disponibilidade e autenticidade, apresentados a seguir.

a) **Confiabilidade**: A confiabilidade trata de garantir que todos os dados e informações, serão somente acessados aos usuários permitidos, e que cada um possa utilizar ou modificar esses arquivos, dentro do seu limite estabelecido. Esse acesso é na maior parte das vezes controlado por um usuário e senha de acesso.
b) **Integridade**: A integridade é a garantia que os arquivos ou dados, não formam manipulados, mantendo assim, sua exatidão e certeza da informação trazida e consultada.
c) **Disponibilidade**: A disponibilidade refere-se a certeza de acesso as informações a qualquer tempo.
d) **Autenticidade**: Na autenticidade, deve-se garantir que os dados e informações sejam verdadeiros, ou que aquele usuário de acesso é legitimo.

3.2 VULNERABILIDADE

De acordo com Mcclure, Scambray e Kurtz (2014), a vulnerabilidade na segurança da informação, é definida como qualquer erro, falha, brecha, pontos fracos em um *software* ou em sistema, que um atacante identifique e explore, para se aproveitar e utilizar desta forma, para invadir aquele *software* ou sistema.

Os autores definem como as principais vulnerabilidades de segurança são, senhas fracas, *softwares* desatualizados, vazamentos de informações, *firewalls* mal configurados, servidores de internet mal configurados e faltas de políticas de segurança, entre outros.

3.3 AMEAÇAS

“Uma ameaça consiste em uma possível violação de um sistema computacional e pode ser acidental ou intencional”. (PINHEIRO, 2007, p. 2-3).

Uma ameaça acidental, é o tipo de ameaça que não é planejada, que pode ocorrer devido a falhas ou no *hardware* ou no *software*, como por exemplo, um *software* mal desenvolvido, que prejudique uma rede ou um sistema, que sobrecarregue ou torne mais lento. Já a ameaça intencional, como o próprio nome diz, são ataques planejados, com o intuito de apagar ou destruir alguma informação, alteração, roubos, e até revelações de informações confidenciais, podendo ser ataques de *malware*, engenharia social, ataque DoS, entre outros, que serão detalhados posteriormente nesta pesquisa. (PINHEIRO, 2007).

O autor Pinheiro (2007), ainda diz que existem três aspectos básicos que um sistema deve atender para evitar essas ameaças: a prevenção, a detecção e a recuperação.

A prevenção envolve desde ataques físicos, como invasão, por exemplo, até proteção de arquivos, como sistemas de autenticação e antivírus. (PINHEIRO, 2007).

Na detecção, o autor define como, um bom sistema que alerte sobre possíveis invasões, ou procedimentos ou ações estranhas. É recomendado também manter-se uma auditoria periódica do sistema, para estar-se sempre em alerta com falhas ou problemas que o sistema tem, e que devem ser corrigidos.

Já na recuperação, Pinheiro (2007), diz que trata-se de sempre realizar *backup* de informações, para em casos de perca de informações, esteja sempre ao alcance de recuperação. É recomendado um *backup* periódico também, de acordo com o que se encaixe melhor no sistema.

4 TÉCNICAS DE ATAQUES

Existem algumas técnicas de ataques utilizadas por *hackers* para o roubo de informações, contas bancárias, infecções de redes e sistemas, e tudo mais que sua mente criminosa planeje. Neste tópico serão abordados as principais formas de ataques utilizadas, e como se prevenir dessas ameaças.

4.1 ATAQUES PASSIVOS

Segundo Fernandes et al. (2006), ataques passivo é o tipo de ataque, que não chega a interferir em um sistema ou em uma rede, nem altera ou destrói um conteúdo, mas sim, é utilizado para o atacante analisar o tráfego da rede, obter informações, espionar determinada rede, para saber qual seria a melhor forma de se realizar um ataque realmente efetivo para que cause prejuízo para o usuário, ou empresa, ou que para o atacante, cause ganho. Os ataques passivos são difíceis de serem detectados, pois como não causam alteração, ou movimentação, seus traços dificilmente são encontrados.

A principal forma de ataque passivo, é a engenharia social, em que Mitnick (2003) descreve sendo como, a forma de iludir e enganar as pessoas, fazendo ela acreditar em algo que o atacante não é, com isso, obter informações com ou até mesmo sem o uso mesmo de tecnologia.

Ela busca a melhor forma de atacar uma determinada empresa, por exemplo, ganhando a confiança de uma determinada pessoa que não esteja preparada, buscando saber as falhas do sistema ou da rede, para então, realizar efetivamente um ataque de maneira ativa. (MITNIK, 2003).

4.2 PHISHING

Phishing, termo que vem do inglês, *fishing* (pescaria), é a forma de ataque que se utiliza da engenharia social, para roubo de informações ou transmissão de *malware*. A maior parte dos ataques acontecem através do e-mail, onde o atacante envia e-mails falsos, se passando por uma instituição, uma empresa, ou outra pessoa, afim de chamar a atenção, ou a curiosidade do usuário, fazendo clicar em links, que podem ser redirecionados a páginas falsas, ou infectando o computador do usuário com *malware*. (OUCH, 2013).

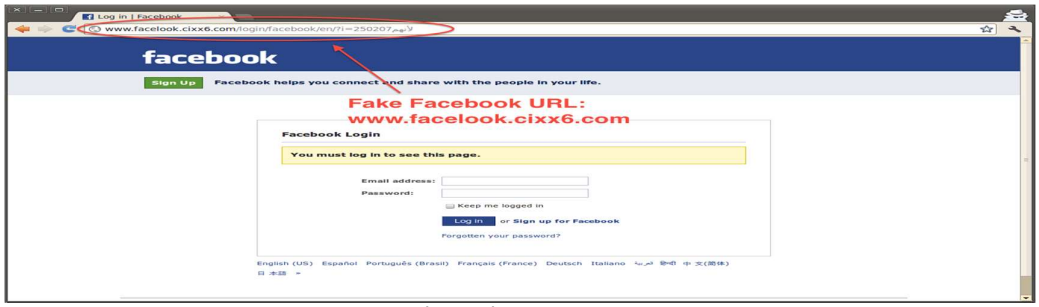
Apesar desta técnica ser mais utilizada através do e-mail, com a popularização das redes sociais, muitos ataques acontecem também através dela, principalmente no Facebook, rede social de maior número de usuários atualmente.

A Figura 1, traz uma imagem de *phising* sendo utilizado através de e-mail, se passando pelo Banco do Brasil, enquanto a Figura 2, traz *pishing* para roubo de senha, através de um site falso do facebook.

Figura 1 – Phishing através de e-mail falso do Banco do Brasil

Fonte: Biasotto (2016).

Figura 2 – Phishing através de um site falso do Facebook



Fonte: Blogwebdesignmicrocamp (2014).

Além desses exemplos apresentados através da Figura 1 e 2, pode-se citar alguns outro clássicos exemplos, como: multa de trânsito, prêmios recebidos, fotos ou informações pessoais divulgadas na internet, etc.

A criatividade dos atacantes é enorme, cada vez mais eles conseguem inventar novas formas de conseguir enganar as pessoas, e o que surpreende, é que grande a quantidade de pessoas que continuam vítimas desse método, mesmo nos dias atuais, com a quantidade de alertas emitidos e repassados para os usuários.

De acordo com uma pesquisa realizada pela Verizon, em 2015, *phishing* continua sendo a maior forma de roubo de dados. Segundo o estudo, de 150 e-mails de *phishing* que foram analisados, 23% desses e-mails são abertos, e 11% são abertos, e são clicados em links ou anexos.

4.3 DENIAL OF SERVICE

O ataque DoS (*Denial of service*, que em português significa, negação de serviço), é uma forma de ataque diferente. Ela não tem como objetivo, a invasão ou a alteração ou destruição, seu objetivo na verdade, como é possível se imaginar pelo nome, é negar, tornar inacessível um serviço, ou uma troca de mensagens, entre outros, e com isso, ela consegue paralisar esta prestação de serviço ou até esgotar esses recursos para a distribuição deste serviço. (LAUFER et al., 2005).

Desta forma, esses autores trazem também, as principais formas que os ataques DoS acontecem:

a) **Ataques por inundação**: Sobrecarga na infraestrutura ou em uma rede, consumindo memória, processamento, espaço em disco. Para que um ataque seja bem sucedido, é necessário gerar um congestionamento maior do que a rede consegue suportar. Sua detecção é complicada, pois é difícil fazer a distinção entre usuário legítimos, e atacantes sobrecarregando a rede. Em redes menores, seu ataque é mais fácil, pois é necessário menor congestionamento, em redes maiores torna-se mais difícil, e para isso é necessário se utilizar o ataque DDoS, ou ataque distribuído.
b) **Ataque por distribuição**: O ataque por distribuição funciona da mesma forma que por inundação, a diferença é que são utilizadas mais estações de ataque, para causar o congestionamento de uma rede maior. Grandes sites já sofreram ataques, como Yahoo, Ebay, Amazon e CNN. Além disso,

existem diversas ferramentas que auxiliam no ataque, tornando menos necessário o conhecimento para realizar os ataques.

c) **Ataque por vulnerabilidade**: Diferente do ataque por inundação, este ataque funciona com a exploração das vulnerabilidades de uma rede, para tornar esses serviços indisponíveis. O ataque funciona, com o atacante descobrindo a vulnerabilidade da rede, e envia um pacote que execute aquelas funções, com isso é possível causar a queda de uma rede, até que se descubra aquele pacote e corrija as vulnerabilidades.

A Figura 3, nos traz um gráfico em que ocorre num ataque DoS, em que o usuário normal, não consegue o acesso, pois o atacante está sobrecarregando um servidor.

Figura 3 – Usuário normal com acesso negado devido à sobrecarga

Fonte: Piratadafaja (2015).

4.4 MALWARE

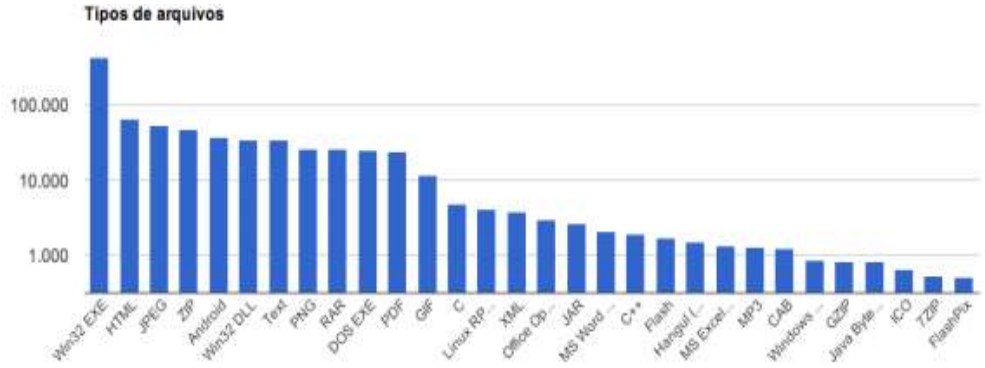
O termo malware, vem do inglês *malicious code* (código malicioso), e é utilizado para todo *software* ou arquivo malicioso que é desenvolvido para ataque de criminosos. Técnicas e ferramentas vem sido empregadas para o combate e a proteção dessas ameaças. (COZZOLINO, 2012).

Davies, Bodmer e Lemasters, (2010), apresentam um estudo da Symantec, que no período entre 2008 e 2009, houveram um crescimento estimado de 1.66 milhão de *malware* neste período. Apesar do estudo ser feito a praticamente um década, com o crescimento da computação, e o maior acesso as pessoas, além do surgimento dos

smartphones, acredita-se que esse número deve crescer a cada ano. Por isso, *malware* é um dos principais assuntos e preocupações quando se fala de segurança no meio da computação.

A Figura 4 nos apresenta uma estatística importante, em que mostra para cada tipo de arquivo, a quantidade de *malware* existente.

Figura 4 - Estatística que mostra a quantidade de malware desenvolvido para cada tipo



Fonte: Vitorino (2013).

Interessante observar neste gráfico, a grande quantidade de *malware* para arquivos *win32.exe*, que é o tipo de arquivo desenvolvido para o Windows, sistema operacional mais popular do mundo. Também é interessante notar que, mesmo o gráfico seja de alguns anos atrás, como existem diversos tipos *malware* para diversos tipos de arquivos, sendo que nenhum seja praticamente livre. Também pode-se imaginar que o número de *malware* tenha crescido desde a época em que a pesquisa foi feita.

Neste tópico, será apresentada sua história, as principais formas de *malware*, seus métodos de proteção e detecção.

4.4.1 História do Malware

Para se saber todas as formas de *malware*, seu uso, é necessário uma breve introdução de seu surgimento.

De acordo com Mell, Kent e Nusbaum (2005), o conceito de vírus já havia surgido desde os primeiros momentos da computação, sendo apenas como brincadeiras e pegadinhas, não para ataques, e crimes.

Os registros dos primeiros *worms* que buscavam o controle de um sistema, é da década de 70, porém o malware não se tornou popular até os meados de 80, onde os cavalos de troia, surgem no meio da década. (MELL, KENT, NUSBAUM 2005).

Já nos anos 90, com a expansão da computação e popularidade de e-mails, o *malware* começou a crescer e se espalhar rapidamente, com a desinformação da maioria das pessoas. (MELL, KENT, NUSBAUM 2005).

Nos anos 2000, apesar dos *worms* serem os mais populares junto com cavalo de troia, surgem outras de formas de *malware*, como os *keyloggers, spyware, adware*, *backdoor, exploits* e *rootkits*. (MELL, KENT, NUSBAUM 2005).

4.4.2 Tipos de Malware

Este tópico é para explicar e mostrar todas as formas de *malware* conhecidas, e a forma que atacam.

4.4.2.1 Vírus de Computador

O famoso vírus de computador, é um programa, que altera, corrompe, destrói, um sistema, podendo-se copiar para outros computadores de uma mesma rede, ou pela internet. (COZZOLINO, 2012).

4.4.2.2 Worms

Os *worms*, parecem com o vírus, mas a diferença entre eles, é que ele é capaz de copiar-se automaticamente, e mais rapidamente. (COZZOLINO, 2012).

4.4.2.3 Cavalo de Troia

O termo cavalo de troia, vem da Grécia antiga, onde troianos tomaram um estrutura de cavalo como um símbolo de vitória, em uma batalha contra os gregos. Porém, o que eles não contavam, é que dentro, haviam soldados gregos, que pegaram o povo troiano desprevenido e facilitando a entrada do exército grego.

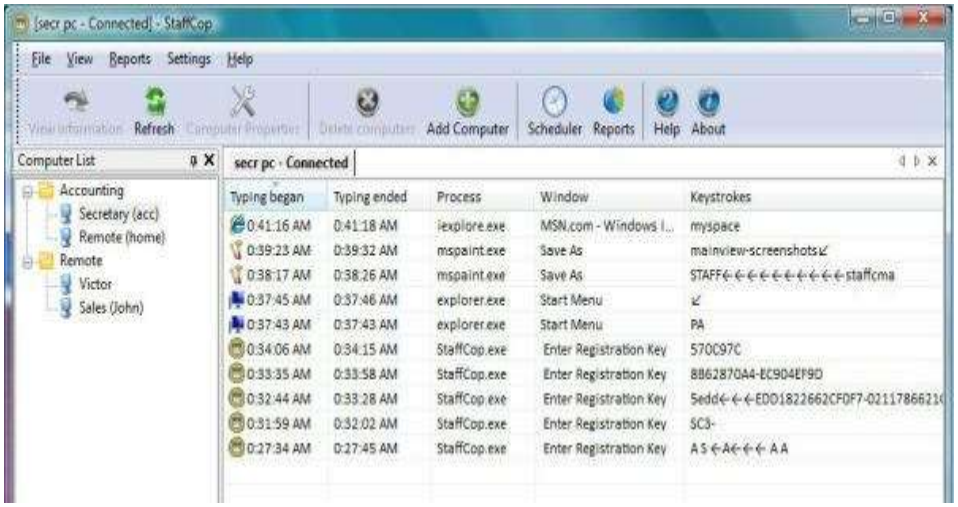
Dessa lenda, vem-se o termo que é usado para definir este tipo de *malware*. Os cavalos de troia, são disfarçados de *softwares* legítimos, porém eles servem para abrir um caminho para um ataque. (COZZOLINO, 2012).

4.4.2.4 Keylogger

Keyloggers são aplicativos utilizados para captação de teclas digitadas, com o objetivo de descobrir senhas, número de cartões, mensagens privadas, entre outros. Eles ficam sendo executados em segundo plano, sem conhecimento do usuário, que assim acaba digitando normalmente sem saber que está sendo tudo registrado. (COZZOLINO, 2012).

A Figura 5, nos apresenta o que seria o registro de um *keylogger*, mostrando a data de início e de fim da digitação, aonde e o que foi digitado.

Figura 5 – Exemplo de relatório de um Keylogger



Fonte Jonathan Machado (2012).

4.4.2.5 Spyware

Como dito pelo nome (*spy*, do inglês espião), é o tipo de *malware* que registra todas as suas atividades realizadas no computador, para a coleta de informações confidencias e dados pessoais. (COZZOLINO, 2012).

4.4.2.6 Adware

Adware é provavelmente o tipo mais irritante, pertubador de *malware*. É aquele que fica exibindo centenas de propagandas em todos os sites que são acessados, deixando a conexão mais lenta. (COZZOLINO, 2012).

A Figura 6 mostra um computador com muitas dela de propaganda, que além de irritantes, tornam o computador sobrecarregado e mais lento.

Figura 6 – Pop-ups de propaganda exibidos por um *adware*

Fonte: Pctools (2011).

4.4.2.7 Backdoor

De acordo com Melo et al. (2010), o *backdoor*, é o *malware* que possibilita o invasor voltar a atacar um computador, ou sistema já comprometido. Quando incluso, o *backdoor* permite que acessos futuros poderão ser feitos remotamente e com uma maior facilidade, sem que seja necessário realizar todo os passos para a invasão.

Os autores Melo et al. (2010), o *backdoor* pode ser realizado como:

a) Zombies: *Zombie*, ou *bot*, é quando um programa que está instalado em um sistema, é utilizado para atacar outro sistema. A maioria de seus ataques são de DDoS. (MELO et al., 2010).

b) Ferramentas de Administração Remota: Instalado em um sistema, permite que o atacante acesse remotamente, podendo verificar todas as ações que estão acontecendo, podendo ligar webcam, o microfone, a qualquer momento sem o conhecimento da vítima. (MELO et al., 2010).

4.4.2.8 Exploits

Um *exploits* é uma sequência de passos e etapas de um *software*, ou uma sequência de códigos, a fim de explorar as vulnerabilidades de um computador, ou sistema. (COZZOLINO, 2012).

Diferente dos outros tipos de *malware*, um *exploit* não precisa necessariamente de um *download* de um arquivo infectado, ou um clique em um determinado link, ele pode dar ao atacante um controle do sistema, permitindo a execução de comandos, e pode também realizar através dele um ataque de DoS.

4.4.2.9 Rootkit

Seu nome surge do Unix e do Linux, onde kits de programas permitindo seu acesso livre a todo conteúdo disponível de todas as camadas, e root vem do termo que é utilizado para o usuário que possui o controle total de uma máquina. Com a sua junção, o atacante pode ter controle total a tudo disponível, para realizarem o que desejarem. (COZZOLINO, 2012).

Assim como praticamente todos os tipos de *malware*, o *rootkit* pode ser infectar um computador através de e-mails, ou de *downloads* de arquivos, ele pode realizar diversas funções, entre as principais, ele utiliza o *keylogger*.

4.4.2.10 Ransomware

De acordo com McDonald e O'Gorman (2012), o *ransomware* é uma categoria de *malware*, que quando o executa, interrompe as funcionalidades de um computador. O *ransomware* faz aparece uma tela com uma mensagem pedindo um pagamento em dinheiro para poder liberar o computador e suas funcionalidades. O *ransomware* é um *malware* que extorque dinheiro dos usuários.

A Figura 7 traz um exemplo típico de *ransomware*, em que o atacante pede uma quantidade de dinheiro para poder liberar o computador para o uso novamente.

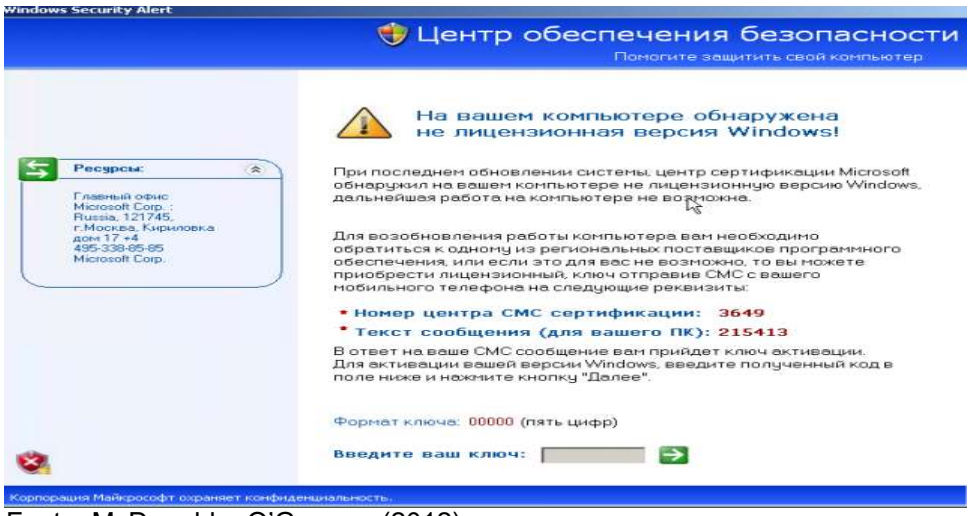
Figura 7 – Exemplo de *Ransomware*

Fonte: McDonald e O'Gorman (2012).

Os mesmos autores nos trazem, que este método de *malware* é novo, sendo visto pela primeira vez na década passada, na Rússia, no ano de 2009, em que o *ransomware* criptografava os arquivos e solicitava pagamento para desencriptação.

A Figura 8 nos traz um dos primeiros *ransomware* criados, na língua russa.

Figura 8 – Exemplo de um dos primeiros *ransomware* russos

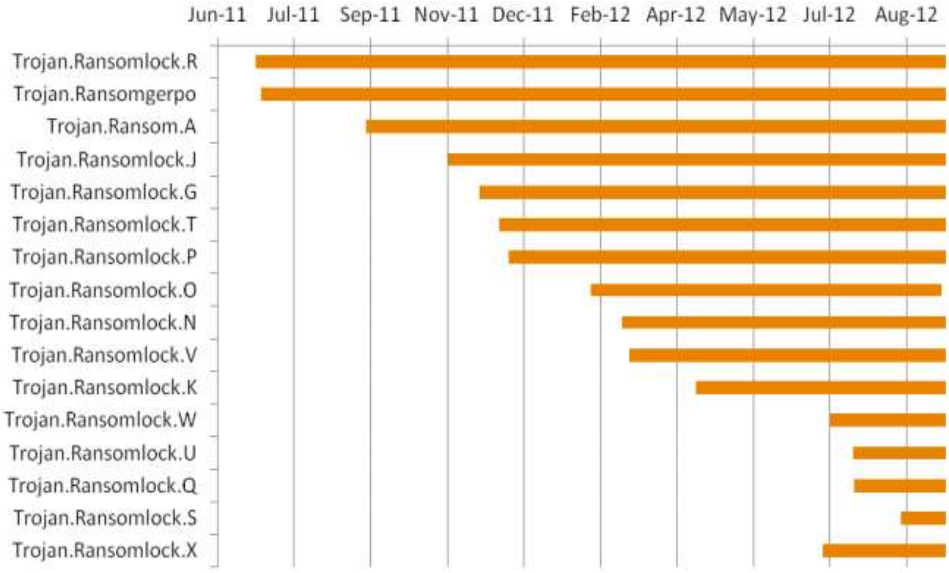


Fonte: McDonald e O'Gorman (2012).

O curioso da figura 8, é que trata-se de uma mensagem suposta da Microsoft, que diz que seu computador deve ser ativado antes do uso. A mensagem é complementa falsa, é claro, sendo não relacionado com a Microsoft. (MCDONALD, O'GORMAN, 2012).

A Figura 9, nos traz um gráfico de atividade de *ransomware* através do uso de trojans, durante o período de junho de 2011, a agosto de 2012.

Figura 9 – Atividade do *rasomware* com uso de *trojans*

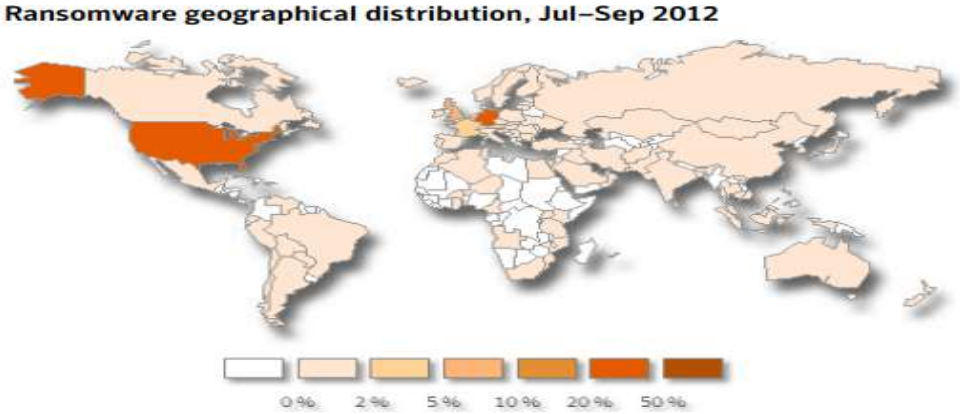


Fonte: McDonald e O'Gorman (2012).

Este gráfico da Figura 9, nos mostra como cada vez surgem mais *ransomware*, e que todos continuam em uso e ataque.

A Figura 10 mostra-nos como o *rasomware* é distribuído geograficamente durante o período de julho a setembro de 2012.

Figura 10 – *Ransomware* distribuído geograficamente



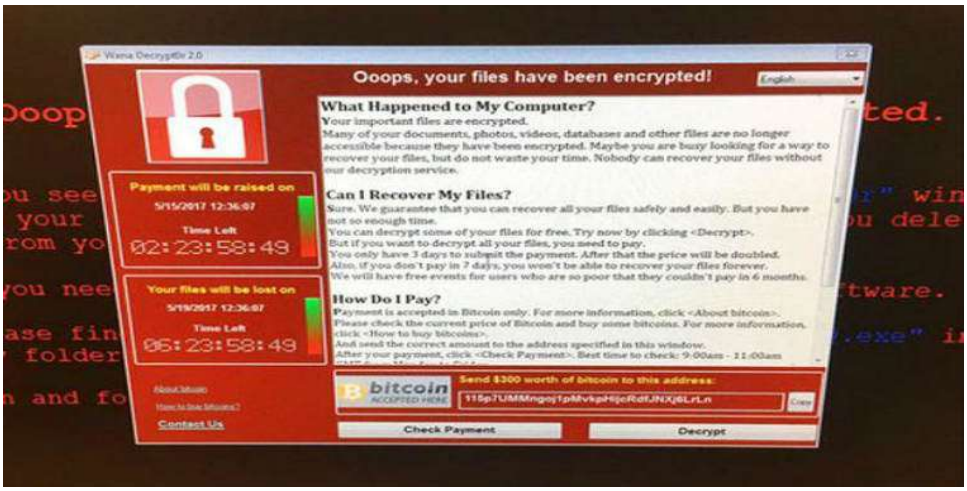
Fonte: McDonald e O'Gorman (2012).

4.4.2.10.1 Ransomware de ataque global

No mês de maio de 2017, um grande ataque global ocorreu de *ransomware*, onde foram afetados mais de 200 mil organizações em 150 países. Conhecido pelo nome de WannaCry, ele bloqueou os dados e conteúdo do computador, e ainda prometia apagar tudo se não for feito um pagamento em bitcoin. (MCGOOGAN, TITCOMB, KROL, 2017).

A Figura 11 apresenta a tela do Wannacry, solicitando pagamento por bitcoin, informando que se feito o pagamento, todos os dados serão deletados, e um cronômetro em regressiva mostrando quanto tempo resta para realizar o pagamento.

Figura 11 – Ransomware WannaCry



Fonte: Nandagopal Rajan (2017).

4.4.3 Evolução

Cada vez mais perigosos, e mais difíceis de serem detectados, os desenvolvedores utilizam algumas técnicas, que são chamadas de técnicas de ofuscação, para não serem detectados por soluções de segurança baseadas em assinatura. Estas técnicas estão juntas na evolução do malware. (COZZOLINO, 2012).

4.4.3.1 Malware Cifrados

O *malware* cifrado, é um método avançado que os desenvolvedores utilizam para cifrarem seu códigos, no intuito de escondê-los. Desta forma, a cada vez que um *malware* se espalhar e infectar outro computador, será diferente, porém realizará as

mesma funções. Apesar de a técnica ser avançada, é vulnerável, pois um antivírus consegue achar o código do *malware* já carregados na memória, e também é possível detecta-lo pela busca de assinatura, pois sua sub-rotina pode ser cifrada. (COZZOLINO, 2012).

4.4.3.2 Malware Oligomórfica

Na técnica de ofuscação oligomórfica, ela aprimora do *malware* cifrado, onde o algoritmo vai variar a decifragem do *malware*. Utilizando várias *engines* de cifragem, escolhe-se assim uma aleatória em cada propagação, sendo assim, a maior parte das *engines* permanecerá cifrada e apenas a correta precisará estar exposta. (COZZOLINO, 2012).

4.4.3.3 Malware Polimórfico

De acordo com o autor Cozzolino (2012), esta foi a primeira técnica que elevou um grau de complexidade na detecção de *malware*. Como no *malware* cifrado, o polimórfico é proliferado através de copias cifradas, que mais tarde retornam a forma original através do módulo de decodificação. A diferença, é que este módulo também é modificado a cada proliferação, sendo assim, se um *malware* polimórfico for bem escrito, sua detecção será bem complexa. Para ser detectado, o *malware* polimórfico, é simulado em um sistema operacional virtual, ou analisando os padrões estático de seu corpo.

Alguns desses códigos maliciosos, utilizam desta técnica para restringir e atrasar sua mutação de forma significativa. Sua vantagem de se modificar mais lentamente, é que desta forma torna-se a abstração de amostras em menores quantidades, não tornando representativa, e assim, *malware* polimórfico pode ser passado despercebido. (COZZOLINO, 2012).

4.4.3.4 Malware metamórfico

O autor Cozzolino (2012) explica que, o *malware* metamórfico, vem para corrigir um ponto fraco do polimórfico, em que seu corpo principal permanece sempre o mesmo a cada geração. Com isso, caso ele seja decifrado, pode ser mais fácil ser detectado baseado em seus padrões. Por isso, o metamórfico é o próximo passo na

evolução, já que ele não depende do processo de cifragem. Ele transforma seu código, modificando registradores, alterando posição de blocos de código, modificando instruções por outras que realizem o mesmo processo, ou seja, basicamente ele se modifica todo, porém irá cumprir toda sua função igual a sua geração anterior.

Além de serem diferentes em discos, eles também ficam na memória do computador, coisa que as outras técnicas de ofuscação não conseguem realizar. (COZZOLINO, 2012).

4.4.4 Análise e Detecção

Neste tópico, será falado sobre como é realizada uma análise de *malware*, e quais as suas metodologias utilizadas.

4.4.4.1 Engenharia Reversa

O autor Eilam (2005) define a engenharia reversa como, fundamental ferramenta e técnica para o entendimento do que um *software* pode fazer. Ela nos permite ver sua estrutura, seu modo de operar, e suas características. A engenharia reversa permite ser realizada de dois modos, manual e automática. A manual é realizada por um programador, e a automatizada, quando o *software* tenta identificar uma estrutura do *malware*.

O autor ainda fala que, a engenharia reversa pode ser utilizada em várias áreas além de *malware*, como na criptografia, na perícia forense, entre outras. Ela é utilizada para buscar e encontrar uma informação escondida, ou em alguns casos, alguma informação que tenha sido destruída, ela é utilizada para tentar recuperar o máximo possível.

Na área de *malware*, a engenharia reversa é definida por duas metodologias, a análise estática, e análise dinâmica.

4.4.4.2 Análise Estática

Sobre análise estática, Cozzolino (2012) diz, “técnica utilizada para coletar informações sobre o programa sem a necessidade de executá-lo.”

A análise estática pode ser feita de várias maneiras, cada uma de preferência e de acordo com nível de conhecimento de cada um. Cozzolino (2012), apresenta

uma forma de ser realizada através de um *disassembler*, que converte o código binário para código assembly. É possível se realizar a análise estática através de um decompilador, que tenta reverter o processo compilado, para uma linguagem de alto nível. Porém, a análise estática necessita de um conhecimento maior e um entendimento sobre o programa, em relação a análise dinâmica, pois o fluxo da execução apenas se baseia no código do programa. O mesmo autor define que os *disassembler* possuem duas divisões, a varredura linear a transversal.

Na varredura linear, o *disassembler* começa pelo início do código binário até seu final, transformando em código assembly, porém isso pode levar a um erro, já que na linguagem assembly tem as instruções misturadas com parâmetros. Caso o ponto de início de uma instrução estiver errado, interpreta-se um parâmetro como instrução, e assim gerando uma cadeia de instruções interpretadas erroneamente. (COZZOLINO, 2012).

Já o transversal, busca corrigir este problema. Caso ocorra de possuir dados no meio do fluxo de instruções, o *disassembler* irá acompanhar este salto, e assim não interpretará erroneamente estes dados. Contudo, este processo realizado estaticamente é complexo, pois este desvio depende dos registradores ou das posições da memória, e saber desta informação estaticamente é difícil, porque não se sabe onde este valor foi carregado. Pode ter sido um pouco antes, um pouco depois, ou muito afastado, além de poder ser resultado de outras operações. (COZZOLINO, 2012).

As técnicas de ofuscação causam grande dificuldade na análise e interpretação a estes *disassemblers* que buscam interpretar automaticamente o código do *malware*.

Os autores Sikorski e Honig (2012), apresentam também uma outra forma de análise, em que ela se baseia por uma certa divisão de tarefas, em que cada uma parte será responsável pela verificação de um ponto, como por exemplos, compressão do código, bibliotecas utilizadas, análise de *strings*, quando e em qual linguagem foi compilado, entre outros. Está forma foi a escolhida e considera a mais eficiente para o desenvolvimento desta pesquisa, que será descrita e detalhada na metodologia.

4.4.4.3 Análise Dinâmica

De acordo com Cozzolino (2012), a análise dinâmica, que diferente da estática, ocorre quando o *malware* está em execução.

O autor Cozzolino (2012), diz que para isso, utilizam ferramentas que fazem verificação de registradores, memória, conteúdo da pilha, entre outros. É comum este tipo de análise ser executada em um ambiente seguro, em uma máquina virtual onde um sistema operacional é simulado, e caso o *malware* realize ataques, não será atingido o computador, somente aquele ambiente emulado.

4.5 PREVENÇÃO

Quando é se falado em *malware*, é importante ressaltar seus métodos de prevenção, pois caso contrário, o computador estará exposto a sofrer ataques e roubos de dados e informações. Ainda mais importante quando se tratado de uma empresa, que deve ser segura, e trazer segurança também aos seus clientes, pois eles precisam confiar na empresa de que nada indevido será acessado.

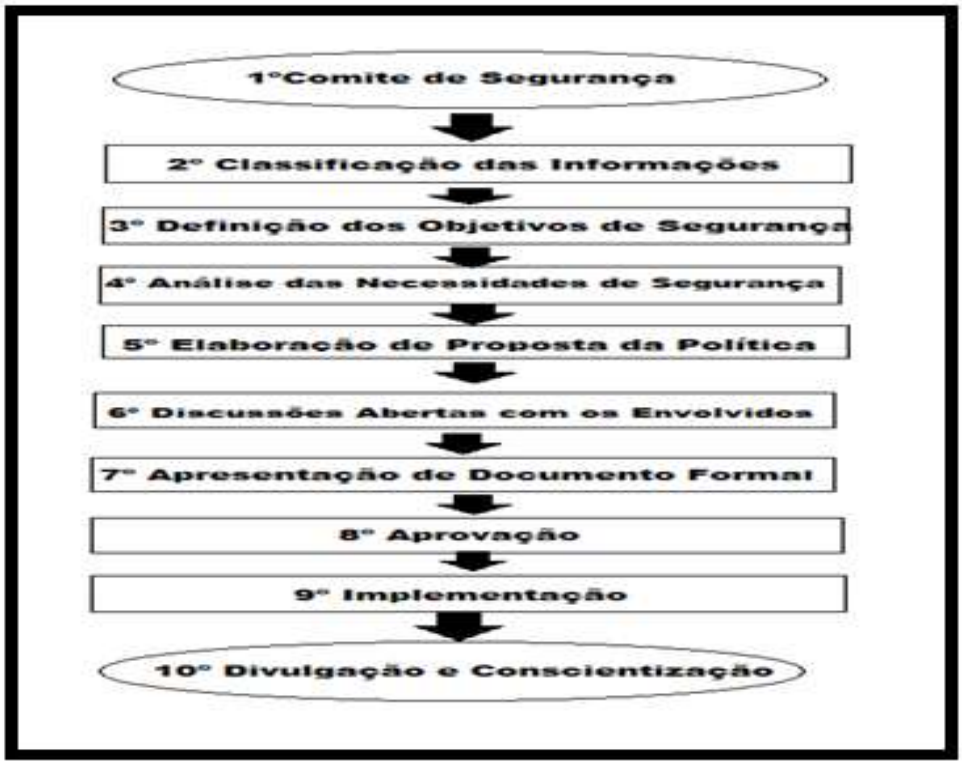
4.5.1 Política de Segurança

Os autores Martins e Santos (2005), definem a política de segurança como um documento que deve descrever as regras, responsabilidades, medidas de segurança de uma empresa. Sendo fundamental o desenvolvimento de uma política de segurança, pois assim todos os funcionários e colaboradores possam garantir os pilares da segurança da informação.

Os autores Martins e Santos (2005) também, ressaltam que não existe uma política de segurança modelo, sendo que ela deve ser estudada caso a caso, e cada uma empresa ou organização possuir uma própria.

A Figura 12 traz um fluxograma de desenvolvimento de políticas de segurança, em que são mostrados passo a passo o processo a ser realizado para se ter uma política de segurança boa, de acordo com Martins e Santos (2005).

Figura 12 – Fluxograma do desenvolvimento das políticas de segurança



Fonte: Martins e Santos (2005).

As características que a política de segurança deve possuir, de acordo com Martins e Santos (2005) são:

a) Propriedade da Informação: Delimitar uma pessoa que fique responsável em controlar o nível de acesso de cada usuário, e definir a periodicidade do *backup*.

b) Controle de Acesso: Todo pedido deve ser documentado, todo usuário deve funções definidas e que não possam realizar outras funções.

c) Gerência de Usuários e Senhas: Senhas únicas, fortes, e individuais, com trocas periódicas.

d) Segurança Física: Controle de entrada e saída de pessoas, de equipamentos, e somente realizado se houver autorização.

A política de segurança, deve ser sempre seguida, e divulgada aos funcionários, para que a segurança possa ser sempre mantida.

4.5.2 Antivírus

O antivírus é a técnica mais comum usada para a proteção, detecção e combate de *malware* e ataques. Nos dias atuais, é fundamental e necessário um antivírus de qualidade em seu computador, devido ao enorme número de *malware.* (MELL, KENT, NUSBAUM, 2005).

Além disso, os autores escrevem sobre as características e qualidades de um bom antivírus, que devem scannear todo o sistema e seus componentes, observação em tempo real de atividades, verificando que não há nada suspeito, monitoração de aplicativos comuns, como os de e-mail, navegadores, entre outros, identificar os tipos comuns de *malware*, além de desinfetar o computador, tornando-o seguro para uso.

4.5.3 Antispyware

De acordo com Mell et. al. (2005), o *spyware* diferente do antivírus, tem como foco arquivos, programas, que tem como objetivo espionar, roubar a privacidade, e dados pessoais. Porém, eles também ressaltam que não só são importantes para isso, como este tipo de *malware*, tende a causar instabilidade e uma performance mais lenta em seu computador.

Com isso, os autores reforçam e recomendam o uso de tanto antivírus como de *antispyware*, pois ambos são importantes para a segurança

4.5.4 Firewall

Segundo Nakamura e Geus (2000, p. 6) “O *firewall* é um elemento essencial na arquitetura de segurança de qualquer organização”. Isto porque, ele atua na borda de uma rede, de um sistema, de um computador, realizando o controle de acesso a realmente a quem deve, e dando a passagem a pacotes seguros. Suas funções podem ser conter um ataque *hacker* bloqueando as entradas, filtrar o tráfego de saída, entre outras. Todo o tráfego passa pelo *firewall*, sempre.

Os autores Nakamura e Geus (2000), completam dizendo os três métodos de existentes de *firewall*, o filtro de pacotes, filtro de pacotes com base no estado da conexão e o filtro de pacotes na camada de aplicação.

De acordo com Nakamura e Geus (2000), o funcionamento do filtro de pacote, acontece com ele permitindo ou negando pacotes de acordo com seu endereçamento ou número de porta.

Os autores Nakamura e Geus (2000), explicam que o *firewall* de filtro de pacotes com base no estado da conexão, atuando a partir de dois elementos, que são, os dados que estão contidos no cabeçalho do pacote e em uma tabela de estados. Esta tabela armazena informações do estado de todas as conexões que trafegam, e o *firewall* com esses dados, baseado em regras pré-definidas, permite ou nega a passagem de um pacote.

Por último, Nakamura e Geus (2000), finalizam dizendo sobre o *firewall* com filtro na camada de aplicação, sendo ele o mais complexo, pois ele utiliza um código para filtrar a aplicação desejada. Este *firewall*, com um bom filtro de aplicação, pode identificar *malware* trafegando, seja ele saindo ou chegando.

5 MÁQUINAS VIRTUAIS

Os autores Laureano, Maziero e Jamhour (2003) citam (POPEK, GOLDBERG 1974), trazem que uma máquina virtual é uma duplicata isolada, e eficiente de uma máquina real. Em uma máquina real, uma camada de baixo nível de *software* fornece o acesso de recursos para o sistema operacional, vindo do *hardware*, disponibilizando de forma abstrata às aplicações. Na máquina real, quando o sistema operacional acessa os dispositivos de *hardware*, ele se utiliza de *device drivers,* que interagem com a memória e os dispositivos de entrada e saída.

Os mesmos autores ainda citam (MALLACH, 1973), afirmando que uma máquina virtual é totalmente oposto, pois ela implementa todas as instruções de uma máquina real em um ambiente abstrato, podendo assim, por exemplo, executar um aplicativo de Windows no Linux, ou vice-versa.

Os autores definem como as vantagens e benefícios sobre o uso de máquina virtual:

a) Aperfeiçoar e testar novos sistema operacionais.

b) Uso para ensino de sistemas operacionais e programação.

c) Execução de diversos sistemas operacionais sobre um mesmo *hardware* simultaneamente.

d) Simular diversas situações que ocorrem em máquinas reais.

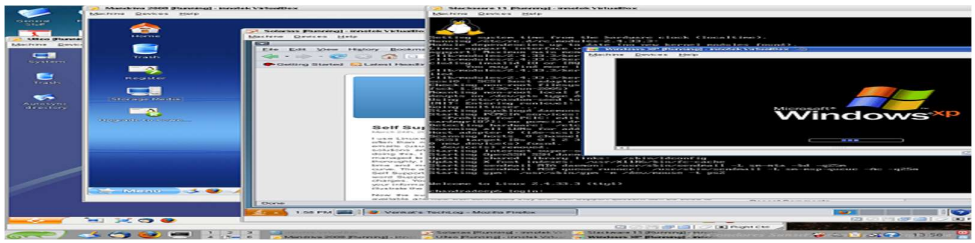
e) Diminuir custos de *hardware*.

f) Desenvolver aplicações com portabilidade para diversos sistemas operacionais.

E como desvantagem do uso, é apenas definido o custo de processos a mais na máquina virtual, podendo ser até de 50% ou até mais, em alguns casos.

A Figura 13 apresenta sistemas operacionais diferentes sendo utilizados sobre uma máquina real, através do uso de máquinas virtuais.

Figura 13 – Uso de máquina virtual para criação de diversos sistemas operacionais



Fonte: Edvaldo Brito (2013).

5.1 WINDOWS

No ano de 1981, a Microsoft começa a desenvolver um gerenciador de interfaces, que viria a ser chamado de Windows, o sistema operacional que se tornaria o de maior uso em todos os computadores. Porém, no seu surgimento, ele ainda não era categorizado como um sistema operacional, que só iria ocorrer no ano de 1993, com o surgimento do Windows NT. O que eles eram considerados antes disso, eram sistemas gráficos que eram executados sobre alguma versão compatível do DOS, como por exemplo, MS-DOS, PC-DOS, ou DR-DOS. (SILVEIRA, 2011).

5.1.1 Windows 7

O Windows 7, sistema operacional escolhido para o desenvolvimento deste trabalho, foi lançado no ano de 2009 e alcançou um enorme sucesso, graças a inovação de seu design, recursos e ferramentas, e possui em duas versões, 32 e 64 bits. Suas qualidades são estabilidade, praticidade, eficiência e clareza, e está na história entre os melhores sistemas operacionais lançados. (LIMA, 2016).

5.2 ANALISE DE MALWARE EM MÁQUINA VIRTUAL

Segundo Laureano et al. (2003), a máquina virtual é importante auxiliar na análise dinâmica e estática de *malware*, pois não torna necessário uma máquina real sem *malware*, para a análise, simples para salvação e restauração de dados, além comparar os estados antes e depois da instrução. Verificar informações de baixo nível, e a tradução dinâmica de instruções pode ser utilizada para instrumentalizar o fluxo de instruções executados pelo *malware*.

6 TRABALHOS CORRELATOS

Na área de detecção de *malware*, podem ser encontrados muitos trabalhos. Por isso, foram separados e apresentados quatro trabalhos encontrados, que são considerados os mais relevantes e no mesmo sentido desta pesquisa.

Melo et al. (2010) trazem a prática de detecção de *malware* estática e dinâmica, além de trazer os traços e evidências, em uma máquina já comprometida. Em sua pesquisa, também é focado apenas em cavalos de troia bancários.

Seu primeiro processo é uma utilização de máquina virtual para realizar as análises. Dividido em dois cenários, sendo o primeiro em uma máquina infectada, e em outra não. Somente no cenário dois, são realizadas os dois métodos de análise, a estática e dinâmica.

Suas conclusões são que, para combater e analisar *malware* é necessário entender todo o processo de análise, e as ferramentas necessárias para detecção.

Também encontrado, Cozzolino (2012), traz em seu trabalho, o problema da detecção do *malware* metamórfico, que é aquele que faz pequenas modificações em código, a fim de não ser detectado. Com isso, ele propõe, através do uso da técnica estática e dinâmica, criar uma metodologia, que a partir de um *malware* conhecido, detectar todas as suas variantes metamórficas.

Cozzolino (2012), utiliza um *disassembler* para conversão do código para *assembly*. Depois, ele acaba dividindo pequenos trechos de código, que serão normalizados e calculados, para a identificação e comparação.

Por trabalhar com *malware* metamórfico, ele realiza diversos testes com um mesmo *malware*, pois a cada teste, ele se transforma em um outro tipo de código. Ele também verifica, quais antivírus do mercado, são capazes de detectar aquele *malware*.

Idika e Mathur (2007), escreveram um artigo cientifico, descrevendo e explicando em três formas de detecção de *malware*, na anômala, de especificação e de assinatura, sendo cada uma delas feitas pelos meio estático, dinâmico e híbrido.

Eles apresentam novas classificações para técnicas de detecção de *malware*, levantam informações e questionam porque existem poucas pesquisas sobre a análise estática em *malware* anômalos, sendo de acordo com eles, a técnica mais promissora e importante. Sugerem também o surgimento de uma técnica universal para que possa realizar em todos tipos de *malware*.

Em outro trabalho correlato encontrado, Tewari, Rastogi e Agarwal (2013) apresentam um estudo sobre as técnicas dinâmicas existentes, e buscam desenvolver um método hibrido, utilizando também a estática, para que seja mais rápida e melhor a detecção.

Em sua pesquisa, eles observam que encontram obstruções para a detecção de *malware* polimórfico, e opacos para detecção através do método estático, necessitando de um método híbrido.

Para o desenvolvimento do método híbrido, ele estudaram o dinâmico e estático e combinaram aonde a semântica é a mesma, mas acessa *loops* ou funções em momentos diferentes. Também foi adicionado saídas prováveis para verificação se é um provável *malware* ou não. Reduzem o tempo da análise dinâmica, selecionando apenas um trecho do código e analisando. Para assim, realizar o método híbrido, e mais eficiente.

Estes foram os que mais chamaram a atenção e o interesse, além de trazerem referências para a pesquisa de desenvolvimento deste trabalho. A área de detecção de *malware* tem grande área de pesquisa e crescimento, pois é sabido que cada vez os *malware* se tornam mais perigosos e mais difíceis de serem encontrados. As técnicas devem sempre estarem sendo aprimoradas, para não ficarem ultrapassadas. Além disso, como visto em trabalhos apresentados, é necessário uma construção de técnica hibrida, que utilize os fortes das duas técnicas, para uma mais rápida e mais completa detecção.

7 METODOLOGIA

O primeiro processo feito, foi realizar um levantamento bibliográfico, para entender o funcionamento das técnicas, as diferente formas de *malware* e de como atuam, planejou como seria feita a análise mantendo a segurança, e definir quais resultados que se esperavam alcançar. Partindo disto, chegou-se à conclusão que para uma maior segurança, a detecção seria executada em um máquina virtual, no sistema operacional Windows 7, 32 bits. Este sistema operacional foi escolhido, devido seu ótimo desempenho junto aos *softwares* que foram utilizados para análise e coleta de resultados.

Para a criação deste ambiente virtual, foi utilizado o *software* VMware Workstation Pro, em um notebook Acer Intel Core I3, com 6 GB de memória RAM, com sistema operacional Windows 10, 64 bits. O ambiente virtual do Windows 7 criado no VMware, utilizou-se de uma memória de 1 GB e com um disco rígido de 60 GB.

Assim que o ambiente virtual foi criado, e suas configurações definidas, conforme mostra a Figura 14.

Figura 14 - Infraestrutura da Máquina Virtual Windows 7

Fonte: Elaborada pelo autor.

Com isso, o primeiro passo foi realizar conexão com internet, para poder obter exemplos que *malware* para os testes e análises, que foram coletados através do site <*http://praticalmalwareanalyis.com/labs//*>, site esse, encontrado na obra de Sikorski e Honig (2012), com um *malware* executável, para realização de testes e análises.

Após obtê-lo e depois de todo conteúdo levantado, verificou-se que para a análise estática, ela deveria ser dividida em cinco partes, sendo: *Hash*, verificação do

arquivo suspeito em sites com banco de dados sobre *malware*, *packers*, *strings* e *PE headers*.

Na etapa de *hash*, em que ela é um identificador único de cada *software*, *malware*, utilizou-se o *software* md5deep, a fim de compara-la com outras *hashs*, e verificar se ela combina com algum tipo de *malware* já encontrado e detectado.

Para a verificação do arquivo em um banco de dados, onde ele verifica se algum antivírus o detecta como *malware*, e também o que ele detecta, utilizou-se o site <*http://virustotal.com/*>.

Em *packers*, ou pacotes em português, o seu objetivo é comprimir o máximo possível o *malware*, para assim dificultar sua análise. Neste processo foi utilizado duas tecnologias, sendo a primeira o *software* PeiD, que busca verificar quais pacotes escondidos e que comprimem o arquivos estão sendo utilizados, e depois o upx, que busca descomprimir o arquivo, a fim de conseguir uma melhor visualização em partes que possam estar comprimidas e escondidas.

Em *strings*, que é um série de caracteres, busca tentar encontrar traços de códigos e comandos, que possam dar dicas, ou pistas, ou pequenas noções do que aquele *software* possa realizar, sempre lembrando que, na análise estática, se verifica o arquivo suspeito sem a sua execução.

E por fim, em *PE headers,* em que pode ser revelado algumas informações, como bibliotecas utilizadas, tipo de aplicação, entre outras. Foi utilizado o *software* Dependency Walker para isso.

Na análise dinâmica, em que diferente da estática, o *malware* é analisado enquanto executa, com isso foram utilizadas algumas ferramentas, sendo elas: Sysinternal Suite, Fakenet e Regshot.

O primeiro passo da análise dinâmica, foi utilizar o Regshot, em que sua função foi de capturar dados e informações do sistema, para depois que o *malware* for executado, capturá-lo novamente, afim de comparar os estados do sistema antes e depois.

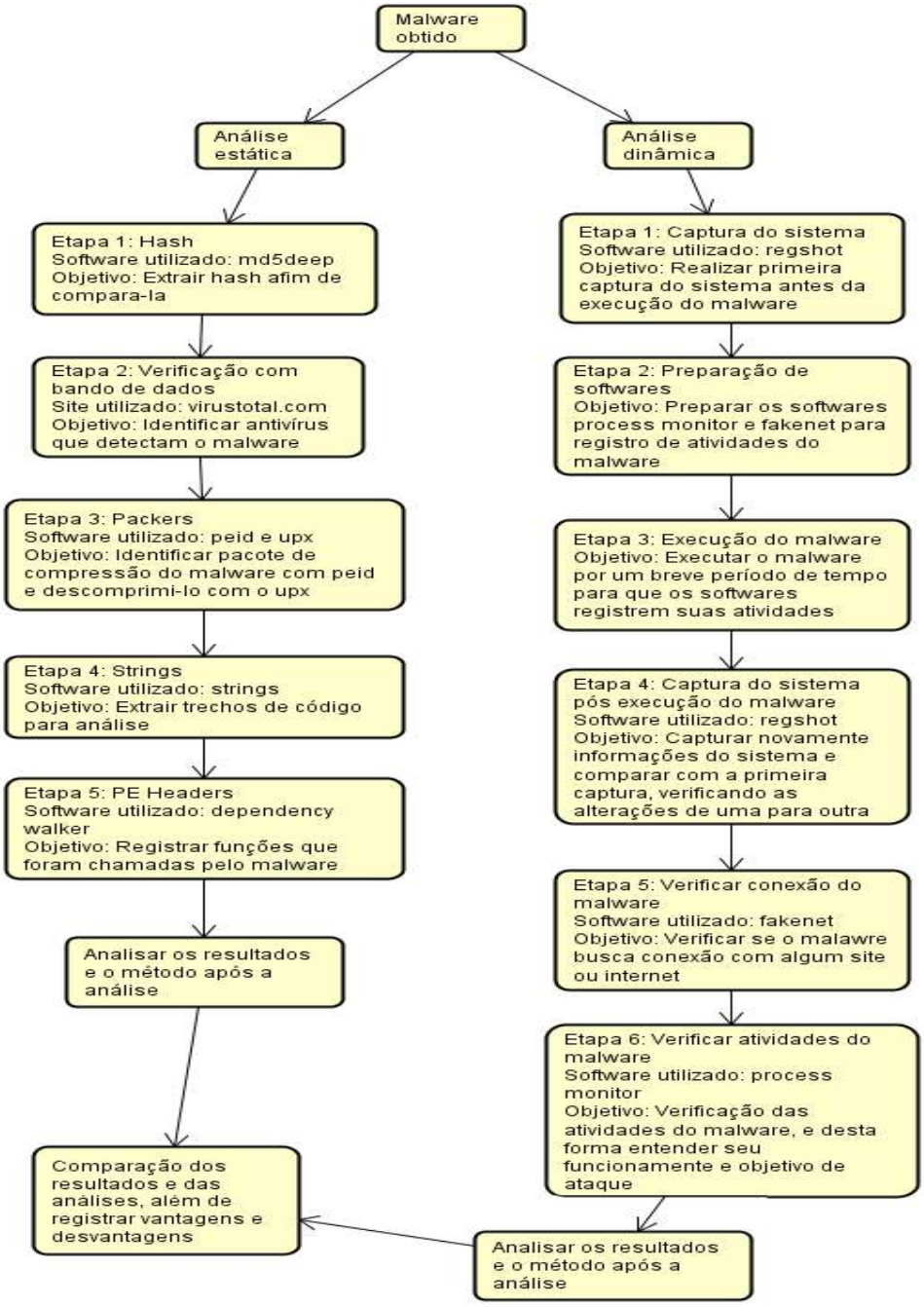
Com este processo executado, pode-se também iniciar alguns *softwares* que se obteve com o Sysinternal Suite, como Process Monitor buscando sempre estar atento com o que o *malware* estava interagindo.

O próximo passo foi utilizar o Fakenet, que é uma ferramenta, que simula uma rede, para que o *malware* interaja com ela, com isso permitindo observar e analisar toda sua atividade em uma rede segura.

Por fim, com todos processos iniciados e preparados, executou-se o *malware*, por um breve período de tempo, e capturou-se novamente todos os dados através do Regshot, e iniciou-se a análise.

A Figura 15 apresenta um diagrama que mostra os softwares utilizados em cada análise e em qual etapa.

Figura 15 – Diagrama da metodologia



Fonte: Elaborada pelo autor.

Após estes processos, o resultado esperado foi de que, tanto no método de análise estática e dinâmica, pudesse entender como cada método funciona, verificando como cada análise atua sobre o *malware*, que tipo de conclusões e resultados pode-se obter a partir de cada um, se é mais completa que a outra, se alguma é mais segura que outra, enfim, dados que levem a entender após a prática e análise aplicada sobre um *malware*.

8 RESULTADOS E DISCUSSÕES

Neste tópico inicia-se todo o desenvolvimento proposto pelos objetivos deste trabalho, baseado na metodologia descrita, iniciando pela análise estática e posteriormente dinâmica. Por fim, um comparativo entre elas.

8.1 ANÁLISE ESTÁTICA

O método de análise estática, como já explicado anteriormente no levantamento bibliográfico, busca identificar elementos, trechos, dados, que levem a conclusão, ou a uma ideia de que aquele arquivo possa ser um *malware*. Como dito na metodologia, a análise estática foi dividida em processos, em que cada um será aprofundado e demonstrado nos tópicos a seguir.

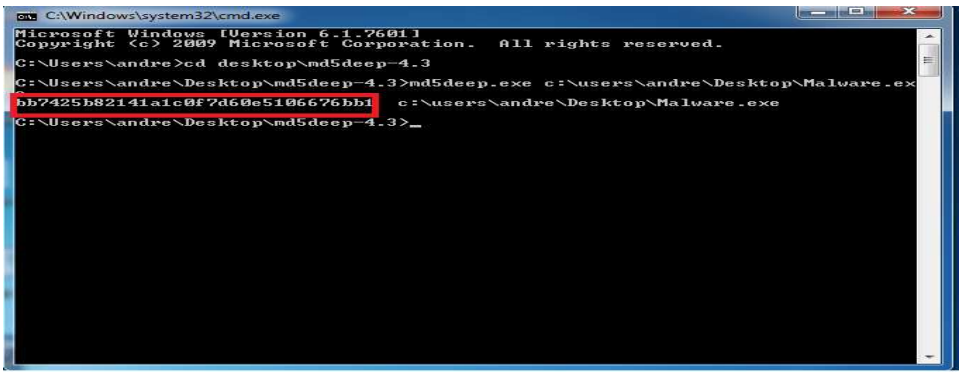
8.1.1 Hash

Iniciou-se a análise pela parte de *hash*, que como dito anteriormente, é um identificador único que cada *software*, que se tem como objetivo obtê-lo e busca-lo, com o intuito de verificar se algum *malware* já identificado possui a mesma *hash*, o que com isso, nos faria saber que, aquele *software* também possa ser um.

Para isso, iniciou-se abrindo o cmd, e foi digitado o comando “cd Desktop\md5deep-4.3” para movimentar-se até a pasta do md5deep. Após mover-se para a pasta, se deu o comando “md5deep.exe C:\Users\Andre\Desktop\Malware.exe”, que levou a execução do *software* e desta forma pode-se obter a *hash* do arquivo.

A Figura 16, mostra destacado a *hash* encontrada após este processo.

Figura 16 – Hash do arquivo encontrada

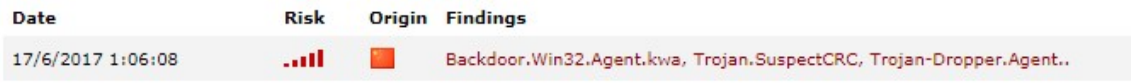


Fonte: Elaborada pelo autor.

Após extrair a *hash*, pode-se verificar em um banco de dados se aquela *hash* é igual outra já encontrada. O banco de dados verificado foi do site <*https://isc.sans.edu/tools/reversehash.html*>.

A Figura 17 nos mostra que a *hash* encontrada foi correspondida na pesquisa com outra igual, e ela aponta pra um *malware*, que foi apontada como um *backdoor*/*trojan* de origem da China.

Figura 17 – Hash encontrada apontando malware



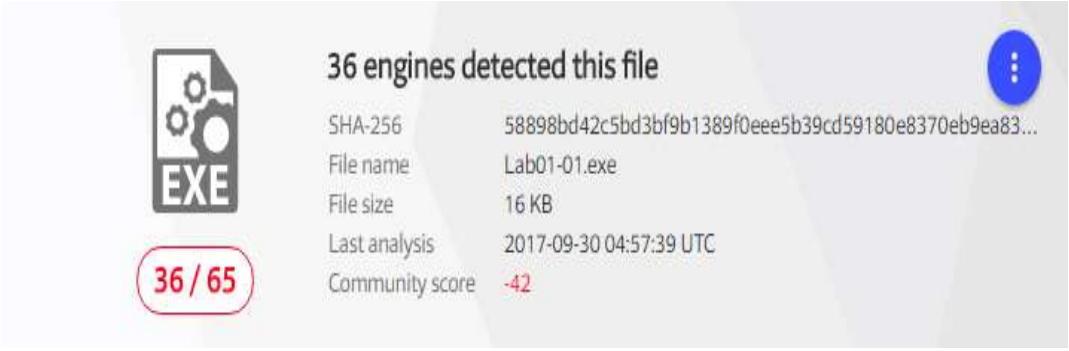
Fonte: Elaborada pelo autor.

Com isso, se finaliza o processo de coleta e verificação da *hash*. Após este processo pode-se tirar de conclusão que a suspeita de que este arquivo que está sendo analisado, pode ser realmente um *malware*, mas para obter maiores conclusões, de não apenas, se ele é ou não, mas como ele atua, e o que mais ele possa conter, deve-se continuar com os processos de análise estática.

8.1.2 Verificando em banco de dados

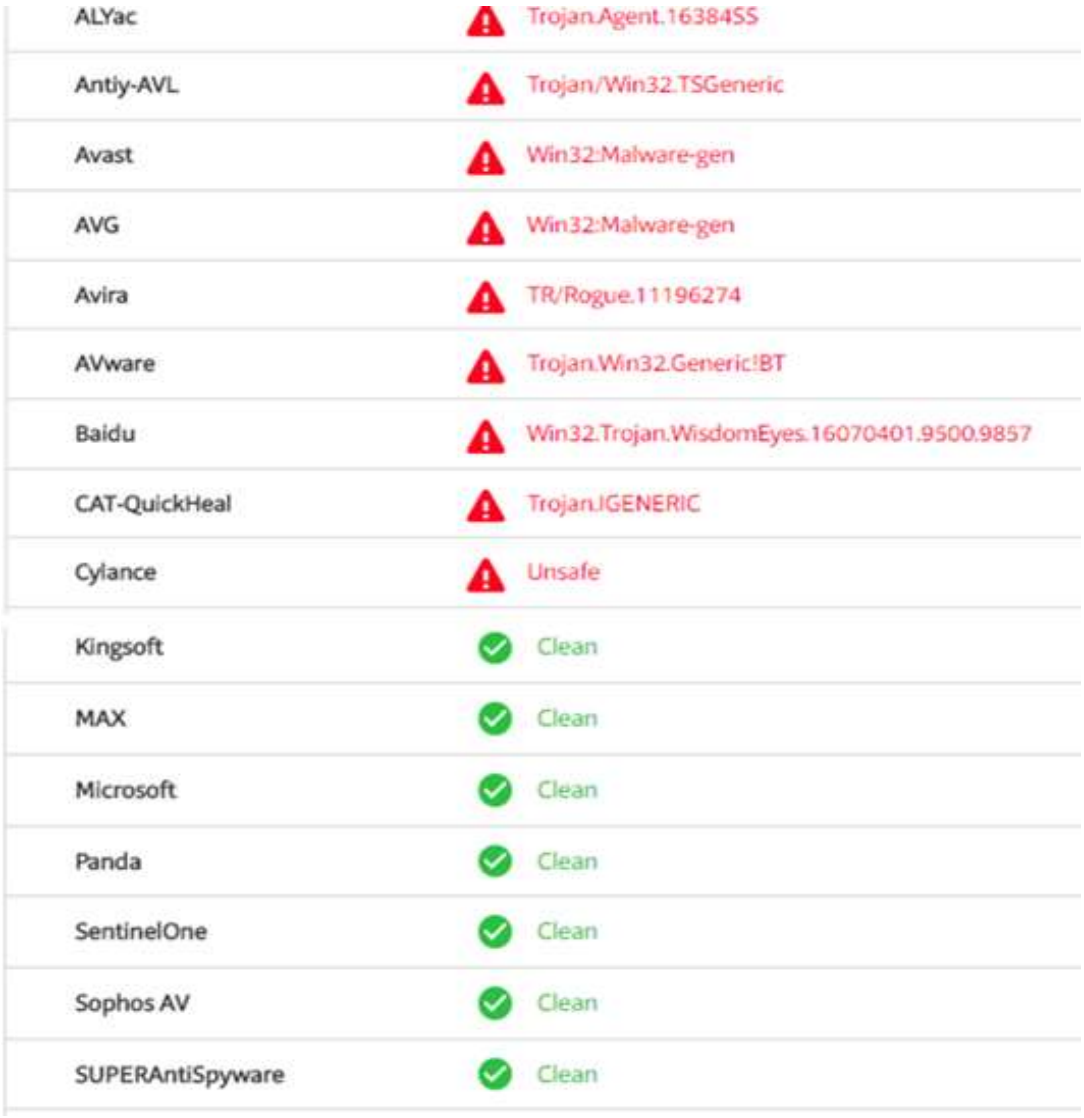
Seguindo a análise, neste processo que chega-se a ser um processo mais simples, em que deve-se carregar o arquivo no site <*https://virustotal.com*> para fazer uma verificação se ele já é identificado por algum antivírus como *malware*. A Figura 18 nos mostra que a taxa de detecção dele em relação aos antivírus é de 36 em 65, ou seja, um pouco a mais da metade, e a Figura 19 nos mostra alguns exemplos de antivírus que o detectam e outros que em que é considerado limpo.

Figura 18 – Taxa de antivírus que detectam



Fonte: Elaborada pelo autor.

Figura 19 – Arquivo detectado por uns e por outros não



Fonte: Elaborada pelo autor.

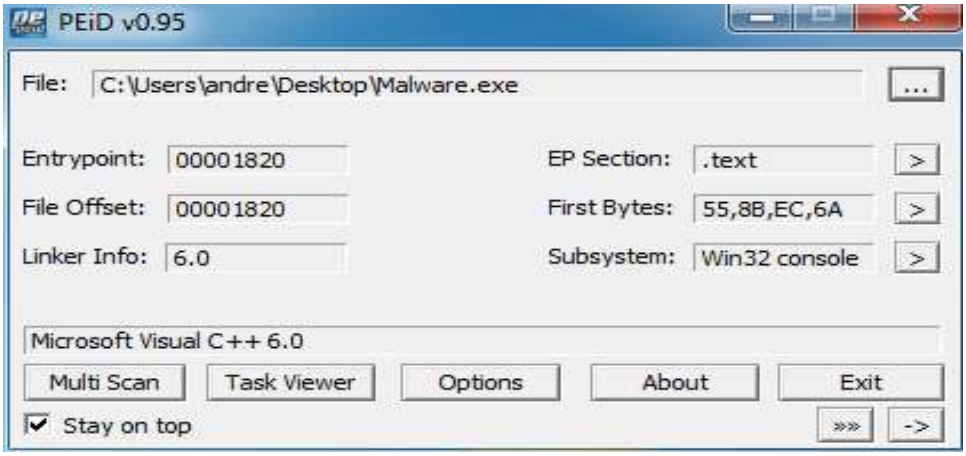
Com isso, conclui-se que nem sempre um antivírus irá identificar todos os *malware*, podendo passar um ou outro. Também pode-se concluir que realmente este arquivo se trata de um *malware*, porém seguimos com a análise estática, para extrairmos mais informações sobre ele.

8.1.3 Packers

No processo de *packers*, em que deve-se verificar se o arquivo está comprimido, e se estiver, é necessário descomprimir, para poder-se seguir com a

análise. Para isso, utilizou-se o *software* Peid, e desta forma, começou-se carregando o arquivo no *software*, e com isso foram conseguidos os resultados de que ele foi compilado em C++ e está com comprimido o pacote da linguagem. A Figura 20 nos mostra a interface do programa com o resultado obtido.

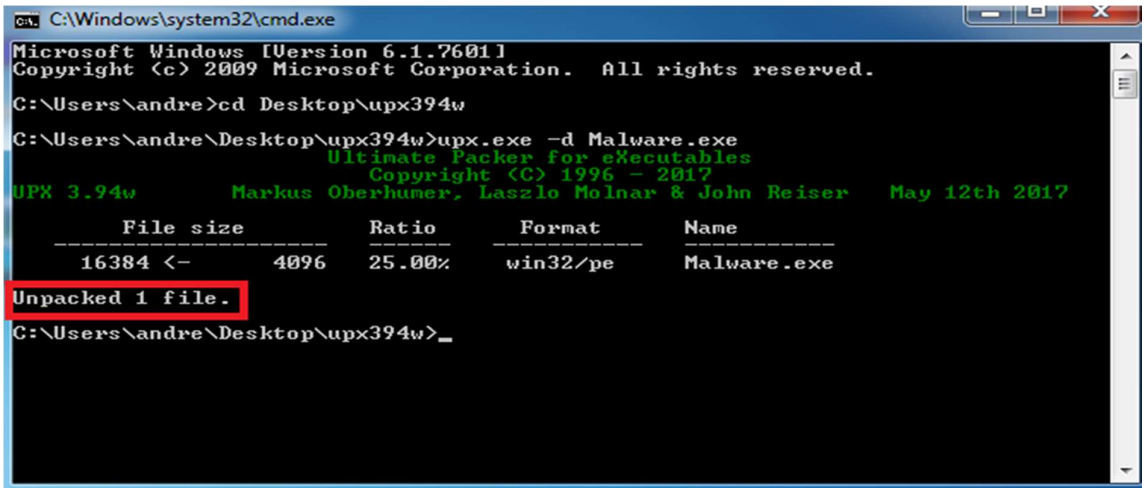
Figura 20 – Pacote do C++ identificado



Fonte: Elaborada pelo autor.

Com isso, deve-se prosseguir o processo e descomprimir o arquivo com o upx. Para isso, deve-se copiar o arquivo para a pasta do executável do upx, abrir o cmd novamente, e mover-se até aquele diretório, onde se deu o comando “upx.exe –d malware.exe” e desta forma se descomprimiu o arquivo, tornando melhor para análise no próximo processo. A Figura 21 mostra o cmd com o comando pra descomprimir o arquivo, e o resultado de sucesso obtido.

Figura 21 – Arquivo descomprimido



Fonte: Elaborada pelo autor.

Com isso, finalizou este processo, que é muito importante para a fase de *strings*.

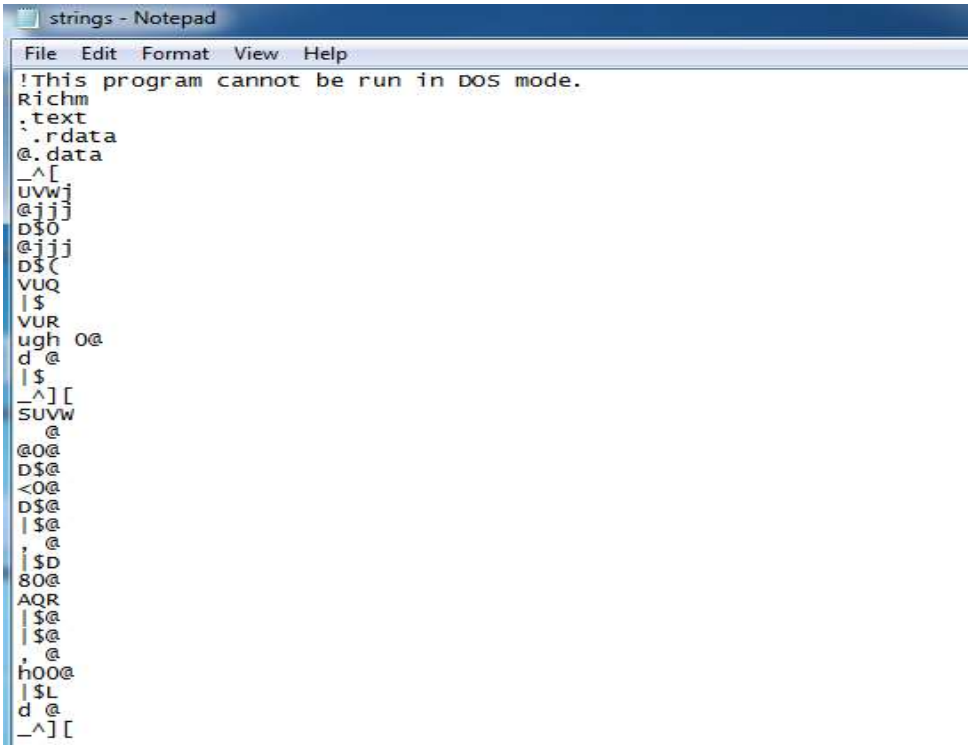
8.1.4 Strings

Em *strings*, busca-se analisar trechos do código para poder compreender o que aquele *software* está buscando, utilizando, qualquer tipo de informação que ajude a ter maior compreensão do que se está analisando.

Assim, começou abrindo cmd novamente, e foi digitado o comando “Strings.exe C:\Users\andre\Desktop\Malware.exe > C:\Users\andre\Desktop\strings.txt”. Com isto, buscou-se executar o Strings no arquivo Malware, e passar todo código encontrado para um arquivo texto, para melhor visualização.

Lembrando que este processo se busca algumas informações relevantes e que sejam legíveis, a Figura 22 mostra um trecho do código que deve ser desconsiderado, por ser ilegível, pelo menos nesta fase, pois possa talvez conter alguma encriptação, ou então está simplesmente para confundir e atrapalhar a análise.

Figura 22 – Código ilegível a ser desconsiderado



Fonte: Elaborada pelo autor.

O que é importante retirar de relevante, como mostra a Figura 23, são os dois arquivos DLL (esses arquivos podem conter códigos, dados, e recursos), em que foram analisadas na próxima fase o que elas fazem, mas pode-se saber que certamente não são coisas boas, devido aos resultados encontrados até aqui. Foram encontrados dois arquivos: Kernel32.dll e MSVCRT.dll. Também pode-se observar a DLL do Kernel, alterando na pasta C:\Windows\System32, que é importante para o funcionamento do sistema operacional.

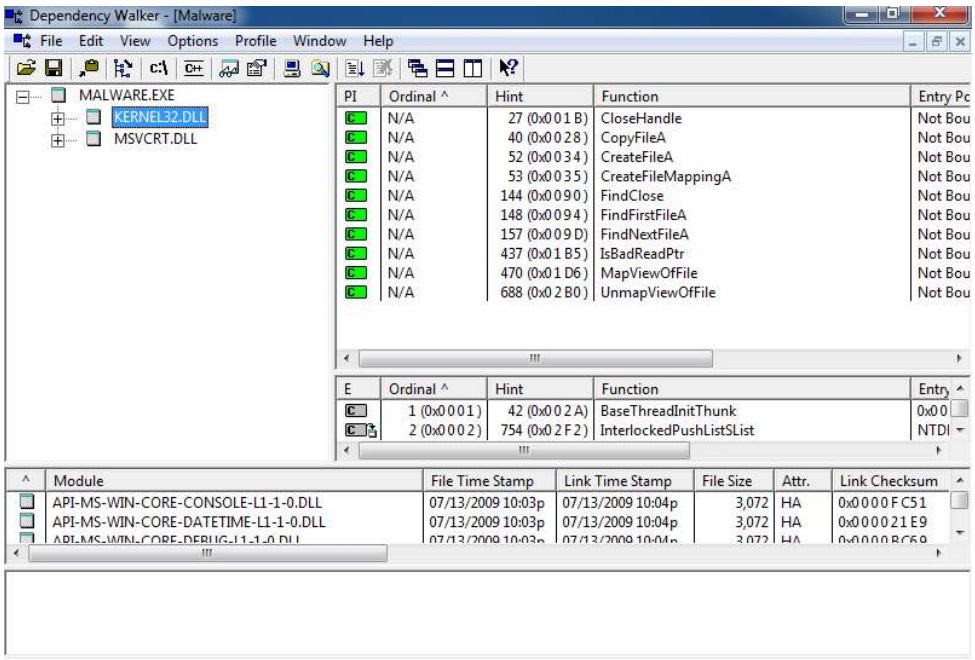
Figura 23 – Arquivos DLL encontrados no código

Fonte: Elaborada pelo autor.

8.1.5 PE Headers

Nesta parte final de análise estática, em que se analisa as funções dos arquivos DLL encontrados no processo anterior, e para isso utilizou o *software* Dependency Walker. O primeiro processo foi carregar o arquivo no software e ele importou as DLLs. A Figura 24 mostra a interface do programa, com o arquivo carregado e as DLLs importadas.

Figura 24 – DLLs importadas do arquivo

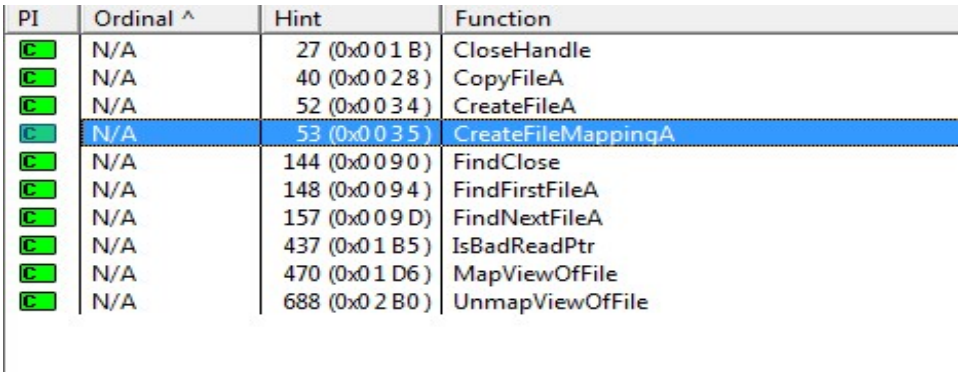


Fonte: Elaborada pelo autor.

Após isto, analisou a DLL Kernel32.dll, por ela ser extremamente importante, pois ele pode conter acesso e manipulação na memória, arquivos e *hardware*.

A Figura 25 mostra as funções que ela está chamando ou utilizando. Como a Microsoft, mantém registrado o que cada função faz, podemos ressaltar as mais importantes como, CopyFileA, CreateFileA, CreateFileMappingA e MapViewOfFile, podemos saber que esta DLL abre um arquivo e o mapeia na memória. FindFirstFilaA e FindNextFileA juntos, podemos observar que eles procuram um arquivos e usa o CopyFileA para copia-los.

Figura 25 – Funções da DLL



Fonte: Elaborada pelo autor.

8.1.6 Discussão dos resultados

Desta forma, encerrou a análise estática e com isso pode-se analisar e discutir os resultados obtidos.

Como já esperado, a análise estática não trouxe muitos resultados profundos, ou seja, não se pode obter muitas informações de como o *malware* atua, ou altera. Conseguiu neste caso, saber que realmente se trata de um *malware*, mas muitas das vezes não é assim, pois cada dia surgem mais *malware* novos, que ainda não foram identificados, e muitas desses processos ele pode se passar limpo. Embora tudo isso, já se pode ter um princípio de ideia do que pode-se encontrar posteriormente na análise dinâmica, e por isso, os dois métodos são tão importantes e se complementam, pois cada um depende do outro para um análise completa.

A Figura 26 apresenta um quadro com as etapas e os resultados obtidos na análise.

Figura 26 – Resultados análise estática

Análise Estática	
Etapa	Resultado
Hash	Com o software md5deep, encontrou a hash e indentificou detectada como malware
Verificação em banco de dados	Arquivo detectado por antivírus como malware
Packers	Com o software Peid, detectou pacote de ofuscação e descompriu
Strings	Codígo análisado, detectado possível alterações em DLL
PE Headers	Funções aplicadas sobre a DLL identificadas e analisadas

Fonte: Elaborada pelo autor.

8.2 ANÁLISE DINÂMICA

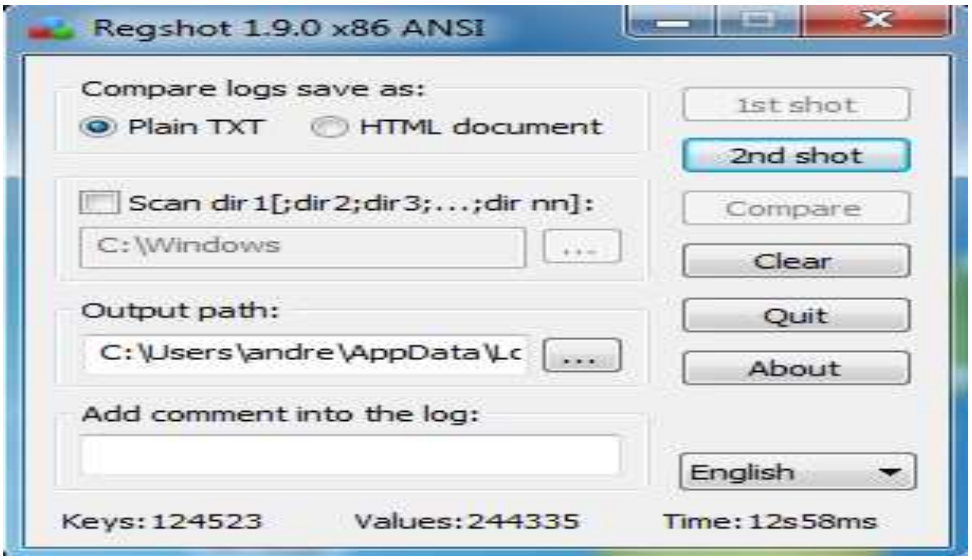
Agora, no método dinâmico, prosseguiu com análise sob o mesmo *malware*, sendo agora mais perigoso e complicado devido a ser necessário maiores cuidados, pois neste processo o *malware*, esteve em execução em todo momento.

8.2.1 Captura das informações do sistema inicial

Nesta etapa, iniciou-se utilizando o Regshot, cujo objetivo foi registrar e capturar informações do sistema sem a execução do *malware*, e após todo processo de análise, se registrou novamente e verificou quais foram suas alterações, para se compreender no que ele atua.

Com isso, a Figura 27, mostra o Regshot com sua primeira captura realizada, esperando a segunda.

Figura 27 – Primeira captura



Fonte: Elaborada pelo autor.

Após este processo, o *malware* foi executado, para análise em outras etapas. No final, será retornado ao Regshot e irá realizar a segunda captura para comparação.

8.2.2 Execução e análise do malware

Com todos as ferramentas já preparadas, chegou-se a hora de executar o arquivo Malware.exe, e deixa-lo executando por um pequeno período de um ou dois minutos, para que os *softwares*, fossem registrando os dados e informações, que depois foi analisada. Após um período aproximado de um minuto, se fez a segunda captura do sistema através do Regshot, para iniciar a análise de todo os conteúdos.

Com a segunda captura, o Regshot registrou um total de 309 mudanças no sistema, sendo 121 valores modificados, 42 adicionados, 76 deletados e 13 chaves adicionadas, conforme mostra a Figura 28.

Figura 28 – Relatório do Regshot

```
Regshot 1.9.0 x86 ANSI
Comments:
Datetime: 2017/10/23 21:14:01   ,   2017/10/23 21:32:34
Computer: WIN-AQ37BP5D1D2 , WIN-AQ37BP5D1D2
Username: andre , andre
Keys added: 13
Keys deleted: 57
Values deleted: 76
Values added: 42
Values modified: 121
Total changes: 309
```

Fonte: Elaborada pelo autor.

Neste relatório, foi interessante reparar como o *malware* deleta arquivos DLL, e cria outros no lugar na pasta System32, pasta esta fundamental para o sistema operacional. Lembrando que DLL, é uma biblioteca dinâmica que podem conter códigos, recursos. Com o *malware* adicionando DLLs próprias, ele poderia controlar e realizar diversos procedimentos ocultos. A Figura 29 mostra arquivos DLLs modificados pelo *malware*.

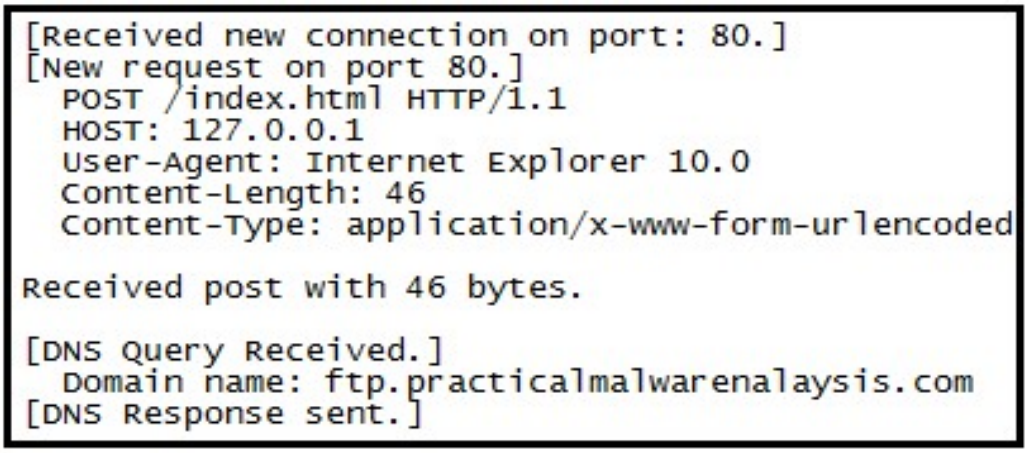
Figura 29 – DLLs modificadas

```
ProtocolName: "@%SystemRoot%\System32\msvcrt.dll,-60100"
PackedCatalogItem:  25 53 79 73 74 65 6D 52 6F 6F 74 25 5C 73 79 73 74 65 6D 33 32 5C 6D 73 77 73
 00 00 00 00 00 02 00 00 00 02 00 00 00 10 00 00 00 10 00 00 00 02 00 00 00 11 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0
ProtocolName: "@%SystemRoot%\System32\KernelBase.dll,-60101"
PackedCatalogItem:  25 53 79 73 74 65 6D 52 6F 6F 74 25 5C 73 79 73 74 65 6D 33 32 5C 6D 73 77 73
 00 00 00 00 00 02 00 00 00 02 00 00 00 10 00 00 00 10 00 00 00 03 00 00 00 00 00 00 00 FF 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0
ProtocolName: "@%SystemRoot%\System32\ntdll.dll,-60102"
PackedCatalogItem:  25 53 79 73 74 65 6D 52 6F 6F 74 25 5C 73 79 73 74 65 6D 33 32 5C 6D 73 77 73
 00 00 00 00 00 02 00 00 00 17 00 00 00 1C 00 00 00 1C 00 00 00 01 00 00 00 06 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0
ProtocolName: "@%SystemRoot%\System32\locale.nls,-60100"
PackedCatalogItem:  25 53 79 73 74 65 6D 52 6F 6F 74 25 5C 73 79 73 74 65 6D 33 32 5C 6D 73 77 73
 00 00 00 00 00 02 00 00 00 17 00 00 00 1C 00 00 00 1C 00 00 00 02 00 00 00 11 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0
ProtocolName: "@%SystemRoot%\System32\wship6.dll,-60101"
PackedCatalogItem:  25 53 79 73 74 65 6D 52 6F 6F 74 25 5C 73 79 73 74 65 6D 33 32 5C 6D 73 77 73
 00 00 00 00 00 02 00 00 00 17 00 00 00 1C 00 00 00 1C 00 00 00 03 00 00 00 00 00 00 00 FF 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0
ProtocolName: "@%SystemRoot%\System32\wship6.dll,-60102"
PackedCatalogItem:  25 53 79 73 74 65 6D 52 6F 6F 74 25 5C 73 79 73 74 65 6D 33 32 5C 6D 73 77 73
 00 00 00 00 00 02 00 00 00 17 00 00 00 1C 00 00 00 1C 00 00 00 01 00 00 00 06 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0
```

Fonte: Elaborada pelo autor.

Com o FakeNet, pode-se observar o *malware* tentando se conectar através do Internet Explorer, com o *site* <*ftp.practicalmalwarealaysis.com*>, conforme mostra a Figura 30. Com esta conexão, ele poderia estar realizando *downloads* de DLLs, compartilhando dados, entre outros.

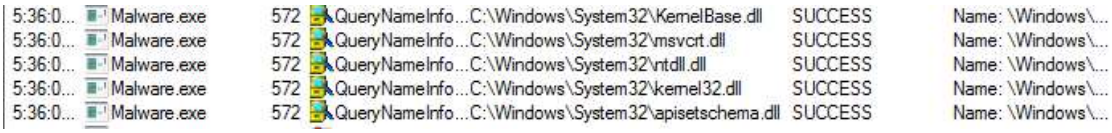
Figura 30 – Malware conectando site



Fonte: Elaborada pelo autor.

Após estas verificações, analisou o que o Process Explorer havia registrado sobre o Malware.exe. A Figura 31, nos mostra que o Malware.exe, através da *QueryNameInformationFile* busca informações e detalhes da DLLs já vista e encontradas na análise estática.

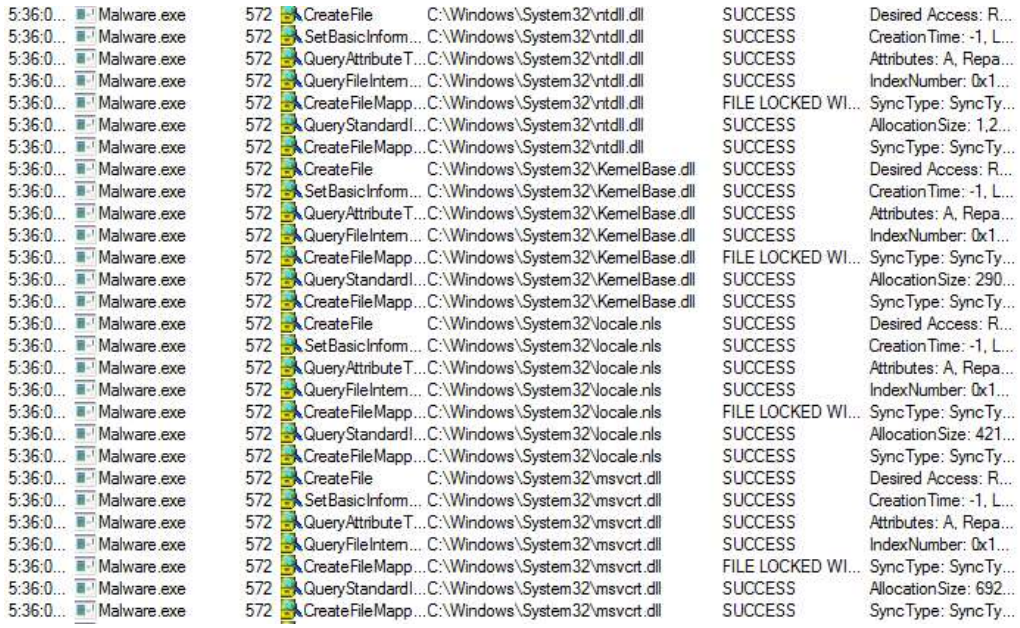
Figura 31 – Malware buscando DLLs



Fonte: Elaborada pelo autor.

Após está busca, o *malware* realiza uma série de processos com as DLLs, o CreateFileMapping, QueryStandardInformationFile, QueryFileInternalInformationFile, QueryAttributeTagFile, SetBasicInformationFile e CreateFile respectivamente, conforme mostra a Figura 32.

Figura 32 – Malware processando DLLs



5:36:0...	Malware.exe	572	CreateFile	C:\Windows\System32\ntdll.dll	SUCCESS	Desired Access: R...
5:36:0...	Malware.exe	572	SetBasicInform...	C:\Windows\System32\ntdll.dll	SUCCESS	CreationTime: -1, L...
5:36:0...	Malware.exe	572	QueryAttributeT...	C:\Windows\System32\ntdll.dll	SUCCESS	Attributes: A, Repa...
5:36:0...	Malware.exe	572	QueryFileIntern...	C:\Windows\System32\ntdll.dll	SUCCESS	IndexNumber: 0x1...
5:36:0...	Malware.exe	572	CreateFileMapp...	C:\Windows\System32\ntdll.dll	FILE LOCKED WI...	SyncType: SyncTy...
5:36:0...	Malware.exe	572	QueryStandardI...	C:\Windows\System32\ntdll.dll	SUCCESS	AllocationSize: 1,2...
5:36:0...	Malware.exe	572	CreateFileMapp...	C:\Windows\System32\ntdll.dll	SUCCESS	SyncType: SyncTy...
5:36:0...	Malware.exe	572	CreateFile	C:\Windows\System32\KernelBase.dll	SUCCESS	Desired Access: R...
5:36:0...	Malware.exe	572	SetBasicInform...	C:\Windows\System32\KernelBase.dll	SUCCESS	CreationTime: -1, L...
5:36:0...	Malware.exe	572	QueryAttributeT...	C:\Windows\System32\KernelBase.dll	SUCCESS	Attributes: A, Repa...
5:36:0...	Malware.exe	572	QueryFileIntern...	C:\Windows\System32\KernelBase.dll	SUCCESS	IndexNumber: 0x1...
5:36:0...	Malware.exe	572	CreateFileMapp...	C:\Windows\System32\KernelBase.dll	FILE LOCKED WI...	SyncType: SyncTy...
5:36:0...	Malware.exe	572	QueryStandardI...	C:\Windows\System32\KernelBase.dll	SUCCESS	AllocationSize: 290...
5:36:0...	Malware.exe	572	CreateFileMapp...	C:\Windows\System32\KernelBase.dll	SUCCESS	SyncType: SyncTy...
5:36:0...	Malware.exe	572	CreateFile	C:\Windows\System32\locale.nls	SUCCESS	Desired Access: R...
5:36:0...	Malware.exe	572	SetBasicInform...	C:\Windows\System32\locale.nls	SUCCESS	CreationTime: -1, L...
5:36:0...	Malware.exe	572	QueryAttributeT...	C:\Windows\System32\locale.nls	SUCCESS	Attributes: A, Repa...
5:36:0...	Malware.exe	572	QueryFileIntern...	C:\Windows\System32\locale.nls	SUCCESS	IndexNumber: 0x1...
5:36:0...	Malware.exe	572	CreateFileMapp...	C:\Windows\System32\locale.nls	FILE LOCKED WI...	SyncType: SyncTy...
5:36:0...	Malware.exe	572	QueryStandardI...	C:\Windows\System32\locale.nls	SUCCESS	AllocationSize: 421...
5:36:0...	Malware.exe	572	CreateFileMapp...	C:\Windows\System32\locale.nls	SUCCESS	SyncType: SyncTy...
5:36:0...	Malware.exe	572	CreateFile	C:\Windows\System32\msvcrt.dll	SUCCESS	Desired Access: R...
5:36:0...	Malware.exe	572	SetBasicInform...	C:\Windows\System32\msvcrt.dll	SUCCESS	CreationTime: -1, L...
5:36:0...	Malware.exe	572	QueryAttributeT...	C:\Windows\System32\msvcrt.dll	SUCCESS	Attributes: A, Repa...
5:36:0...	Malware.exe	572	QueryFileIntern...	C:\Windows\System32\msvcrt.dll	SUCCESS	IndexNumber: 0x1...
5:36:0...	Malware.exe	572	CreateFileMapp...	C:\Windows\System32\msvcrt.dll	FILE LOCKED WI...	SyncType: SyncTy...
5:36:0...	Malware.exe	572	QueryStandardI...	C:\Windows\System32\msvcrt.dll	SUCCESS	AllocationSize: 692...
5:36:0...	Malware.exe	572	CreateFileMapp...	C:\Windows\System32\msvcrt.dll	SUCCESS	SyncType: SyncTy...

Fonte: Elaborada pelo autor.

Começando pelo CreateFileMapping, este processo abre a DLL mapeada pelo Malware.exe, e assim o QueryStandardInformationFile, modifica e define informações na DLL. Depois ele executa novamente o CreateFileMapping para o modo somente leitura, por isso o resultado aparece *File locked with only readers*.

O QueryFileInternalInformationFile, irá consultar e repassar para o *malware* o número de referência do arquivo em byte, e o QueryAttributeTagFile consulta os atributos da DLL, e por fim o SetBasicInformationFile verifica as informações da DLL, para o CreateFile criar um novo arquivo DLL no lugar do anterior, sem parecer que houveram mudanças nele.

Com isso, chegou-se ao fim a análise dinâmica, e podemos observar que detalhes encontrados na análise estática realmente se realizaram, como modificação nas bibliotecas DLL, comprovando realmente se tratar de um arquivo malicioso.

A Figura 33 apresenta um quadro com os *softwares* utilizados na análise dinâmica e seus resultados.

Figura 33 – Resultados análise dinâmica

Análise Dinâmica	
Software	**Resultados**
Regshot	**Com uma captura antes e outra depois da execução, identificou 309 alterações, incluindo nas DLLs, encontradas já na análise estática**
Fakenet	**Identificada conexão com site de origem**
Process Monitor	**Registrou as funções utilizadas para alterações nas DLLs**

Fonte: Elaborada pelo autor.

8.3 COMPARATIVO

Com as análises finalizadas, pode-se comparar os resultados, vantagens e desvantagens, e relação entre elas, sendo tudo isso, demonstrado na Figura 34, para uma melhor e mais clara visualização.

Figura 34 – Comparativo das análises

Análise Estática	Análise Dinâmica
O método estático apresenta uma análise menos profunda, que apresenta caracteristicas do malware, como linguagem compilada, funções, pacotes de ofuscação. Pode ser considerada um primeiro passo pra análise dinâmica, pois com suas informações obtidas, se consegue definir objetivos para serem alcançados na análise dinâmica.	**O método dinâmico apresenta uma análise mais profunda que a dinâmica, pois com ela consegue identificar as alterações que o malware faz em um sistema. Necessário maiores cuidados, pois o risco de ataque, ou até de problemas na coleta de resultados é maior, devido sua execução. Os resultados complementam junto com os resultados obtidos na análise estática.**
Vantagens: Menores riscos e método menos complexo	**Vantagens: Compreensão maior das funções do malware, resultados mais significantes**
Desvantagem: Resultados menos significativos	**Desvantagens: Risco maior de infecção e método mais complexo**

Fonte: Elaborada pelo autor.

9 CONCLUSÃO

Com a finalização deste trabalho, ficou demonstrado como funciona um processo de análise de um *malware*, e o que mais é importante ressaltar que este não é um processo linear e nem padrão. Podendo ser executado de formas diferentes, com ferramentas diferentes, que irão funcionar melhor em cada caso, dependendo sempre do conhecimento e método de preferência do analisador, gerando assim, diversas visões sobre um *malware*.

Também importante se ressaltar que, quando se deseja realizar uma análise, é preciso entender a não se prender a pequenos detalhes, pois um *malware* em maior parte, é grande, complexo, e ofuscado. Neste trabalho foi realizado em um mais simples, mas claro que em ameaças reais, é necessário se analisar de forma mais geral, para se entender seus processos.

Conforme os objetivos estabelecidos no trabalho, pode-se realizar os dois métodos de análise, demonstrando seu funcionamento, e observando suas diferenças, além de concluir que um método complementa o outro para uma análise mais completa. Também foram atingidos os objetivos do levantamento bibliográfico, que deram base e referências para o estudo das técnicas e delimitação dos *softwares*, e processos da análise, além de entender como age e funciona um *malware*.

Por fim, concluo dizendo que está é uma análise menos profunda, um pouco mais simples, para se entender e demonstrar os princípios de uma análise de *malware*. Um próximo passo, para um analisar experiente, seria utilizar-se de disassemblers e debuggers para análise do código em assembly do *malware*, executando e analisando linha por linha de seu código para todo seu entendimento.

Análise de *malware* é um processo que se necessita paciência, segurança, conhecimento, dedicação, sendo fundamental para a segurança da informação.

REFERÊNCIAS

ALEXANDRIA, João Carlos Soares de. **Gestão da Segurança da Informação – Uma Proposta para Potencializar a Efetividade da Segurança da Informação em Ambiente de Pesquisa Cientifica.** São Paulo: 2012. Disponível em:<http://www.iaea.org/inis/collection/NCLCollectionStore/_Public/43/130/43130675.pdf>. Acesso em: 19 abr. 2017.

BIASOTTO, Abner. **Phising.** Disponível em: <http://abnerbiasotto.com.br/artigos/?tag=phishing>. Acesso em: 02 jun. 2017.

BLOG WEB DESIGN MICROCAMP. **Phising - tome cuidado**. Disponível em: <http://blogwebdesignmicrocamp.com.br/mercado-de-trabalho/phishing-tome-cuidado/>. Acesso em: 02 jun. 2017.

BRITO, Edvaldo. Techtudo. **Vmware ou virtualbox: qual o melhor programa para criar máquina virtual?**. Disponível em: <http://www.techtudo.com.br/artigos/noticia/2013/05/vmware-ou-virtualbox-qual-o-melhor-programa-para-criar-maquina-virtual.html>. Acesso em: 02 jun. 2017.

COZZOLINO, Marcelo Freire. **Detecção de Variantes Metamórficas de Malware por Comparação de Códigos Normalizados**. Brasília: 2012. Disponível em: <http://repositorio.unb.br/bitstream/10482/10421/3/2012_MarceloFreireCozzolino.pdf>. Acesso em: 09 maio. 2017.

FERNANDES, Jorge Henrique Cabral. **Gestão da segurança da informação e comunicações**. 1 ed. Brasília: Faculdade de Ciência da Informação, 2010.

FERNANDES, N. C. et al. **Ataques e Mecanismos de Segurança em Redes Ad Hoc**, Rio De Janeiro: 2006. Disponível em: <http://s3.amazonaws.com/academia.edu.documents/32145793/ceseg_2006-sbseg-mc2.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1493932968&Signature=Rh%2Bku3Et8UT6q3jATKcubUwUtpg%3D&response-content-disposition=inline%3B%20filename%3DAtaques_e_Mecanismos_de_Seguranca_em_Red.pdf>. Acesso em: 04 maio 2017.

IDIKA, Nwokedi; MATHUR, Aditya. **A survey of malware detection techniques.** West Lafayette: 2007 Disponível em: <https://www.researchgate.net/publication/229008321_A_survey_of_malware_detection_techniques>. Acesso em: 10 abr. 2017.

LAUREANO, Marcos Aurelio Pchek; MAZIERO, Carlos Alberto; JAMHOUR, Edgard. **DETECÇÃO DE INTRUSÃO EM MÁQUINAS VIRTUAIS.** Curitiba: 2003. Disponível

em:
<https://www.researchgate.net/profile/Carlos_Maziero/publication/228977968_Deteccao_de_intrusao_em_maquinas_virtuais/links/0a85e539b24b73b304000000.pdf>. Acesso em: 19 maio 2017.

LIMA, Davi de. TECHTUDO. **Windows 7**. Disponível em: <http://www.techtudo.com.br/tudo-sobre/windows-7-starter.html>. Acesso em: 21 maio 2017.

MACHADO, Jonathan. Tecmundo. **O que é keylogger?** Disponível em: <https://www.tecmundo.com.br/spyware/1016-o-que-e-keylogger-.htm>. Acesso em: 17 maio 2017.

MARCIANO, João Luiz Pereira. **Segurança da Informação - uma abordagem social**. Brasília: 2006 Disponível em: <http://repositorio.unb.br/bitstream/10482/1943/1/Jo%C3%A3o%20Luiz%20Pereira%20Marciano.pdf>. Acesso em: 19 abr. 2017.

MARTINS, Alaíde Barbosa, Celso Alberto Saibel Santos. **UMA METODOLOGIA PARA IMPLANTAÇÃO DE UM SISTEMA DE GESTÃO DE SEGURANÇA DA INFORMAÇÃO**. Revista de Gestão da Tecnologia e Sistemas de Informação, Salvador: 2005.

MCCLURE, Stuart; SCAMBRAY, Joel; KURTZ, George. **Hackers expostos**: **Segredo e Soluções para a Segurança de Redes**. 7 ed. Porto Alegre: Bookman, 2014.

MCDONALD, Geoff, O'GORMAN, Gavin. **Ransomware: A Growing Menace**. Mountain View: 2012. Disponível em: <http://www.01net.it/whitepaper_library/Symantec_Ransomware_Growing_Menace.pdf>. Acesso em: 22 maio 2017.

MCGOOGAN, Cara, TITCOMB, James, KROL, Charlotte.THE THELEGRAPH. **What is wannacry and how does ransomware work?** Disponível em: <http://www.telegraph.co.uk/technology/0/ransomware-does-work/>. Acesso em: 22 maio 2017.

MEL, Peter; KENT, Karen; NUSBAUM, Joseph. **Guide to Malware Incident Prevention and Handling.** Gaithersburg: 2005. Disponível em:< http://tim.kehres.com/docs/nist/SP800-83.pdf> Acesso em: 17 maio 2017.

MELO, L. P. D. et al. **Análise de Malware: Investigação de Códigos Maliciosos Através de uma Abordagem Prática.** Brasília: 2010 Disponível em: <http://www.peotta.com/sbseg2011/resources/downloads/minicursos/90650.pdf>. Acesso em: 19 abr. 2017.

MITNICK, Kevin; SIMON, WILLIAM L**. A arte de enganar**. 1 ed. São Paulo: MAKRON, 2013.

NAKAMURA, Emilio Tissato; GEUS, Paulo Lício De. **Análise de Segurança do Acesso Remoto VPN.** Campinas: 2005. Disponível em: <https://www.researchgate.net/profile/Emilio_Nakamura/publication/267363565_ANALISE_DE_SEGURANCA_DO_ACESSO_REMOTO_VPN/links/57bae83e08ae3b9d9b1c538d.pdf>. Acesso em: 04 maio 2017.

PC TOOLS. **What is adware and spyware?**. Disponível em: <http://www.pctools.com/security-news/what-is-adware-and-spyware/>. Acesso em: 02 jun. 2017.

PINHEIRO, José Maurício Dos Santos. **Ameaças e Ataques aos Sistemas de Informação: Prevenir e Antecipar.** 2007. Disponível em: <http://web.unifoa.edu.br/cadernos/edicao/05/11.pdf>. Acesso em: 19 abr. 2017.

PIRATA DA FAJÃ. **Como fazer um ataque dos na linha de comandos**. Disponível em: <http://piratadafaja.blogspot.com.br/2011/11/como-fazer-um-ataque-dos-na-linha-de.html>. Acesso em: 02 jun. 2017.

RAJAN, Nandagopal. The Indian Express. **Wannacry ransomware: everything to know about the global cyberattack**. Disponível em: <http://indianexpress.com/article/technology/tech-news-technology/wannacry-ransomware-attack-how-cyber-criminals-can-lock-you-out-of-your-computer-4655910/>. Acesso em: 02 jun. 2017.

R. P. L. et al. **Negação de Serviço: Ataques e Contramedidas**. Rio de Janeiro: 2005. Disponível em: <http://professor.ufabc.edu.br/~joao.kleinschmidt/aulas/seg2013/negacao.pdf>. Acesso em: 09 maio 2017.

SIKORSKI, Michael; HONIG, Andrew. **Practical malware analysis: The Hands-On Guide to Dissecting Malicious Software**. 1 ed. San Francisco: No Starch Press, 2012.

SILVEIRA, Richard Batista. **História do Microsoft Windows**. Cornélio Procópio: 2011. Disponível em: <http://ccp.uenp.edu.br/centros/d_matematica/jcoelho/txt/ap-jc01-windows.pdf>. Acesso em: 21 maio 2017.

TECMUNDO. **Phishing continua sendo maior causa de roubos de dados, segundo pesquisa.** Disponível em: <https://www.tecmundo.com.br/seguranca-de-dados/78315-phishing-continua-sendo-maior-causa-roubos-dados-segundo-pesquisa.htm>. Acesso em: 09 maio 2017.

TECMUNDO. **O que é spyware?** Disponível em: <https://www.tecmundo.com.br/spyware/29-o-que-e-spyware-.htm>. Acesso em: 17 maio 2017.

TEWARI, Ayush; RASTOGI, Ishan; AGARWAL, Shubanshu. **Malware Detection Techiniques.** Índia: 2013. Disponível em: <https://web.iiit.ac.in/~ayush.tewari/RIS.pdf> Acesso em: 17 maio 2017.

VITORINO, Thiago Arreguy Silva. **Análise de Malware em Ambiente Windows**. Brasília: 2013. Disponível em: <https://repositorio.ucb.br/jspui/bitstream/10869/2692/1/Thiago%20Arreguy%20Silva%20Vitorino.pdf>. Acesso em: 17 maio. 2017.

ANÁLISE DAS TÉCNICAS ESTÁTICA E DINÂMICA PARA DETECÇÃO DE MALWARE

André Luís Sanches Benini[1]. Henrique Pachioni Martins[2]. Elvio Gilberto da Silva[2]. Renan Caldeira Menechelli[2].

[1]Graduando em Ciência da Computação pela Universidade do Sagrado Coração (USC) – andreluis_benini@hotmail.com
[2]Centro de Ciências Exatas – Universidade do Sagrado Coração (USC) - henmartins@gmail.com; egsilva@usc.br; renan.menechelli@gmail.com

RESUMO

Na área da segurança da informação, muitas são as ameaças existentes, sendo a maior delas, em termos de quantidade e periculosidade, o malware. Este termo utilizado para códigos maliciosos de uma maneira geral, cujo os objetivos são destruir, alterar, espionar, clonar, sobrecarregar, entre outras funções. Por isso, torna-se fundamental a detecção do malware, para não houver prejuízos. Desta forma, sua detecção é realizada através de análises, uma estática, em que não se executa o objeto que está sendo analisado, outra dinâmica, que realiza durante sua execução. Com este motivo, este trabalho se propôs a realizar e demonstrar como estas análises funcionam e são importantes para a segurança da informação, identificando e demonstrando seus processos, sempre buscando adquirir e passar informações e conhecimentos, uma outra visão, buscando o crescimento e a contribuição com a segurança da informação. Assim, a metodologia aplicada, foi devido ao levantamento bibliográfico, realizando análise estática e dinâmica sobre um malware em um ambiente virtual seguro criado, extraindo, compreendendo, discutindo os resultados conseguidos através desta pesquisa e análises. Com isso observou todo processo de análise é complexo, que requisita paciência em uma análise de malware, sendo observado que, método o estático acaba sendo um pouco menos profundo que o dinâmico, porém, para uma análise completa, uma junção dos dois para se entender melhor o funcionamento, ameaças, e como combater o malware.

Palavras-chave: Malware. Segurança da informação. Análise estática. Análise dinâmica.

1 INTRODUÇÃO

Este trabalho, tem como foco a área de segurança da informação, de forma mais precisa, em *malware*, que são *softwares* maliciosos, construídos para roubo de informações e dados, derrubar ou prejudicar sistemas, invasão de privacidade entre outros. A cada dia, milhares de *malware* são desenvolvidos, cada vez mais perigosos e mais difíceis de serem detectados. Para sua detecção, existem dois métodos:

estático e dinâmico. De forma sucinta, a diferença entre elas é, enquanto a estática analisa o *malware* sem a sua execução, a dinâmica o faz executando.

Neste trabalho foi analisado as formas de detecção de um *malware*, seus métodos de processo e análise, categorizar suas diferenças, através de, implementações em uma máquina virtual. Segundo Sikorski e Honig (2012, p. 28) “A análise de *malware* é a arte de dissecar o malware para entender como funciona, como identificá-lo, e como derrotá-lo ou eliminá-lo”. Sua análise é importante processo na segurança da informação, pois é nela em que é obtido informações de onde ele surgiu, qual seu objetivo de ataque, saber o dano causado por ele, e de que forma se prevenir. Além disso, com sua identificação, se torna mais fácil e rápido encontrar variações e mutações daquele *malware*, facilitando uma detecção futura.

A importância desta pesquisa, é pesquisar e trazer conhecimentos sobre os métodos de detecção, demonstrando seu funcionamento e sua prática, através de realização de testes e analisado *malware*, para que se possa obter e demonstrar os resultados obtidos através destas análises.

Com isso, este trabalho propõe analisar diferentes técnicas de detecção de *malware*, compreender seu funcionamento e a forma que são executadas, além de fazer um comparativo entre elas.

3 SEGURANÇA DA INFORMAÇÃO

A segurança da informação, é a área da computação que trata sobre todos os métodos, técnicas, normas e regras sobre segurança de um sistema, de dados, de arquivos, ou tudo aquilo que é considerado importante e necessário, para uma empresa, ou um usuário comum, com o intuito de protege-lo, tanto virtualmente, mas como também fisicamente.

Para Alexandria (2009), a segurança da informação, significa a proteção da informação do acesso desautorizado, de uma modificação ou destruição.

3.1 PILARES DA SEGURANÇA DA INFORMAÇÃO

Consideram-se os pilares de segurança da informação, a confiabilidade, a integridade, a disponibilidade e autenticidade. (FERNANDES, 2010).

e) **Confiabilidade**: A confiabilidade trata de garantir que todos os dados e informações, serão somente acessados aos usuários permitidos, e que cada um possa utilizar ou modificar esses arquivos, dentro do seu limite estabelecido. Esse acesso é na maior parte das vezes controlado por um usuário e senha de acesso. (FERNANDES, 2010).
f) **Integridade**: A integridade é a garantia que os arquivos ou dados, não formam manipulados, mantendo assim, sua exatidão e certeza da informação trazida e consultada. (FERNANDES, 2010).
g) **Disponibilidade**: A disponibilidade refere-se a certeza de acesso as informações a qualquer tempo. (FERNANDES, 2010).
h) **Autenticidade**: Na autenticidade, deve-se garantir que os dados e informações sejam verdadeiros, ou que aquele usuário de acesso é legitimo. (FERNANDES, 2010).

3.2 VULNERABILIDADE

De acordo com Mcclure, Scambray e Kurtz (2014), a vulnerabilidade na segurança da informação, é definida como qualquer erro, falha, brecha, pontos fracos em um *software* ou em sistema, que um atacante identifique e explore, para se aproveitar e utilizar desta forma, para invadir aquele *software* ou sistema.

Os autores definem como as principais vulnerabilidades de segurança são, senhas fracas, *softwares* desatualizados, vazamentos de informações, *firewalls* mal configurados, servidores de internet mal configurados e faltas de políticas de segurança, entre outros.

3.3 AMEAÇAS

“Uma ameaça consiste em uma possível violação de um sistema computacional e pode ser acidental ou intencional”. (PINHEIRO, 2007, p. 2-3).

Uma ameaça acidental, é o tipo de ameaça que não é planejada, que pode ocorrer devido a falhas ou no *hardware* ou no *software*, como por exemplo, um *software* mal desenvolvido, que prejudique uma rede ou um sistema, que sobrecarregue ou torne mais lento. Já a ameaça intencional, como o próprio nome diz, são ataques planejados, com o intuito de apagar ou destruir alguma informação,

alteração, roubos, e até revelações de informações confidenciais, podendo ser ataques de *malware*, engenharia social, ataque DoS, entre outros, que serão detalhados posteriormente nesta pesquisa. (PINHEIRO, 2007).

4 MALWARE

O termo malware, vem do inglês *malicious code* (código malicioso), e é utilizado para todo *software* ou arquivo malicioso que é desenvolvido para ataque de criminosos. Técnicas e ferramentas vem sido empregadas para o combate e a proteção dessas ameaças. (COZZOLINO, 2012).

4.1 TIPOS DE MALWARE

Este tópico é para explicar e mostrar as principais formas de *malware* conhecidas, e a forma que atacam.

4.1.1 Worms

Os *worms*, parecem com o vírus, mas a diferença entre eles, é que ele é capaz de copiar-se automaticamente, e mais rapidamente. (COZZOLINO, 2012).

4.1.2 Cavalo de Troia

O termo cavalo de troia, vem da Grécia antiga, onde troianos tomaram um estrutura de cavalo como um símbolo de vitória, em uma batalha contra os gregos. Porém, o que eles não contavam, é que dentro, haviam soldados gregos, que pegaram o povo troiano desprevenido e facilitando a entrada do exército grego.

Dessa lenda, vem-se o termo que é usado para definir este tipo de *malware*. Os cavalos de troia, são disfarçados de *softwares* legítimos, porém eles servem para abrir um caminho para um ataque. (COZZOLINO, 2012).

4.1.3 Keylogger

Keyloggers são aplicativos utilizados para captação de teclas digitadas, com o objetivo de descobrir senhas, número de cartões, mensagens privadas, entre outros.

Eles ficam sendo executados em segundo plano, sem conhecimento do usuário, que assim acaba digitando normalmente sem saber que está sendo tudo registrado. (COZZOLINO, 2012).

4.1.4 Backdoor

De acordo com Melo et al. (2010), o *backdoor*, é o *malware* que possibilita o invasor voltar a atacar um computador, ou sistema já comprometido. Quando incluso, o *backdoor* permite que acessos futuros poderão ser feitos remotamente e com uma maior facilidade, sem que seja necessário realizar todo os passos para a invasão.

4.1.5 Exploits

Um *exploits* é uma sequência de passos e etapas de um *software*, ou uma sequência de códigos, a fim de explorar as vulnerabilidades de um computador, ou sistema. (COZZOLINO, 2012).

4.2 DETECÇÃO

Para detecção de *malware*, existem duas técnicas utilizadas, a estática, e a dinâmica, que serão apresentadas nos tópicos seguintes.

4.2.1 Análise Estática

Sobre análise estática, Cozzolino (2012) diz, “técnica utilizada para coletar informações sobre o programa sem a necessidade de executá-lo.”

Os autores Sikorski e Honig (2012), apresentam também uma outra forma de análise, em que ela se baseia por uma certa divisão de tarefas, em que cada uma parte será responsável pela verificação de um ponto, como por exemplos, compressão do código, bibliotecas utilizadas, análise de *strings*, quando e em qual linguagem foi compilado, entre outros. Está forma foi a escolhida e considera a mais eficiente para o desenvolvimento desta pesquisa, que será descrita e detalhada na metodologia.

4.2.2 Análise Dinâmica

De acordo com Cozzolino (2012), a análise dinâmica, que diferente da estática, ocorre quando o *malware* está em execução.

O autor Cozzolino (2012), diz que para isso, utilizam ferramentas que fazem verificação de registradores, memória, conteúdo da pilha, entre outros. É comum este tipo de análise ser executada em um ambiente seguro, em uma máquina virtual onde um sistema operacional é simulado, e caso o *malware* realize ataques, não será atingido o computador, somente aquele ambiente emulado.

5 METODOLOGIA

O primeiro processo feito, foi realizar um levantamento bibliográfico, para entender-se o funcionamento das técnicas, as diferente formas de *malware* e de como atuam, planejar como será feita a análise mantendo a segurança, e definir quais resultados que se espera alcançar. Partindo disto, chegou-se à conclusão que para uma maior segurança, a detecção seria executada em um máquina virtual, no sistema operacional Windows 7, 32 bits. Este sistema operacional foi escolhido, devido seu ótimo desempenho junto aos *softwares* que foram utilizados para análise e coleta de resultados.

Para a criação deste ambiente virtual, foi utilizado o *software* VMware Workstation Pro, em um notebook Acer Intel Core I3, com 6 GB de memória RAM, com sistema operacional Windows 10, 64 bits. O ambiente virtual do Windows 7 criado no VMware, utilizou-se de uma memória de 1 GB e com um disco rígido de 60 GB.

Assim que o ambiente virtual foi criado, o primeiro passo foi realizar conexão com internet, para poder obter exemplos que *malware* para os testes e análises, que foram coletados através do site <*http://praticalmalwareanalyis.com/labs//*>, site esse, encontrado no livro de Sikorski e Honig (2012), com um *malware* executável, para realização de testes e análises.

Após obtê-lo e depois de todo conteúdo levantado, verificou-se que para a análise estática, ele deveria ser dividida em cinco partes, sendo elas: *Hash*, verificação do arquivo suspeito em sites com banco de dados sobre *malware*, *packers*, *strings* e *PE headers*.

Na parte de *hash*, em que ela é um identificador único de cada *software*, *malware*, utilizou-se o *software* md5deep, a fim de compara-la com outras *hashs*, e verificar se ela combina com algum tipo de *malware* já encontrado e detectado.

Para a verificação do arquivo em um banco de dados, onde ele verifica se algum antivírus o detecta como *malware*, e também o que ele detecta, utilizou-se o site <*http://virustotal.com/*>.

Em *packers*, ou pacotes em português, o seu objetivo é comprimir o máximo possível o *malware*, para assim dificultar sua análise. Neste processo foi utilizado duas tecnologias, sendo a primeira o *software* PeiD, que busca verificar quais pacotes escondidos e que comprimem o arquivos estão sendo utilizados, e depois o upx, que busca descomprimir o arquivo, a fim de conseguir uma melhor visualização em partes que possam estar comprimidas e escondidas.

Em *strings*, que é um série de caracteres, busca tentar encontrar traços de códigos e comandos, que possam dar dicas, ou pistas, ou pequenas noções do que aquele *software* possa realizar, sempre lembrando que, na análise estática, se verifica o arquivo suspeito sem a sua execução.

E por fim, em *PE headers,* em que pode ser revelado algumas informações, como bibliotecas utilizadas, tipo de aplicação, entre outras. Foi-se utilizado o *software* Dependency Walker para isso.

Na análise dinâmica, em que diferente da estática, o *malware* é analisado enquanto executa, com isso foram utilizadas algumas ferramentas, sendo elas: Sysinternal Suite, Fakenet e Regshot.

O primeiro passo da análise dinâmica, foi utilizar o Regshot, em que sua função foi de capturar dados e informações do sistema, para depois que o *malware* for executado, capturar-se novamente, afim de compara-los os estados do sistema antes e depois.

Com este processo executado, executou-se também alguns *softwares* que se obteve com o Sysinternal Suite, como Process Monitor buscando sempre estar atento com o que o *malware* estava interagindo.

O próximo passo foi utilizar o Fakenet, que é uma ferramenta, que simula uma rede, para que o *malware* interaja com ela, com isso permitindo observar e analisar toda sua atividade em uma rede segura.

Por fim, com todos processos iniciados e preparados, executou-se o *malware*, por um breve período de tempo, e capturou-se novamente todos os dados através do Regshot, e iniciou-se a análise.

Após estes processos, o resultado esperado foi de que, tanto no método de análise estática e dinâmica, pudesse entender como cada método funciona,

verificando como cada análise atua sobre o *malware*, que tipo de conclusões e resultados pode-se obter a partir de cada um, se um mais completa que a outra, se alguma é mais segura que outra, enfim, dados que levem a entender após a prática e análise aplicada sobre um *malware*.

6 RESULTADOS FINAIS

Iniciou-se o desenvolvimento deste trabalho, com a análise estática, com o processo e verificação da *hash* do *malware*, sendo detectada em um banco de dados de *hashs* como *trojan*. No processo de *packers* identificou que aquele arquivo havia sido compilado na linguagem C++, tendo seu pacote de compressão. Após descomprimir, analisou a *string* gerada, sendo detectada algumas DLLs no *malware*. Com a DLLs encontradas, através do *software* DependecyWalker, detectados que ela atua copiando arquivos, criando arquivos, mapeando, entre outros processos. Após sua finalização, observou-se que a análise estática, não produz resultados profundos e conclusivos, apenas uma ideia e que podemos estar analisando de fato, um arquivo malicioso, partindo assim para a análise dinâmica.

Na análise dinâmica, começou capturando informações do sistema naquele determinado momento, pois após a execução do *malware*, iria se capturar novamente, e verificar quais alterações e adições ou remoções ele realizava. Após um tempo de execução, registrou-se novamente as informações do sistema e realizou um comparação com a passada, registrando um total de 309 alterações. Dentro estas alterações, percebemos modificações em DLLs, como visto anteriormente na análise estática, sendo bibliotecas DLLs fundamentais para o sistema operacional e segurança. Podemos também observar através do Fakenet, que o *malware* tenta conexão com site, podendo realizar *downloads*, compartilhar dados, entre outros. Com o Process Explorer, reparamos que ele realiza uma séries de processos com DLLs, cujo objetivo é modifica-las com conteúdos maliciosos a seu favor.

7 CONSIDERAÇÕES FINAIS

Com a finalização deste trabalho, ficou demonstrado como é funciona um processo de análise de um *malware*, e o que mais é importante ressaltar que este não é um processo linear e nem padrão. Podendo ser executado de formas diferentes,

com ferramentas diferentes, que irão funcionar melhor em cada caso, dependo sempre do conhecimento e método de preferência do analisador, gerando assim, diversas visões e sobre um *malware*.

Também é importante ressaltar que, quando se deseja realizar uma análise, é preciso entender a não se prender a pequenos detalhes, pois um *malware* em maior parte, é grande, complexo, e ofuscado. Neste trabalho foi realizado em um mais simples, mas claro que em ameaças reais, é necessário se analisar de forma mais geral, para se entender seus processos.

Conforme os objetivos estabelecidos no trabalho, pode-se realizar os dois métodos de análise, demonstrando seu funcionamento, e observando suas diferenças, além de concluir que um método complementa o outro para uma análise mais completa. Também foram atingidos os objetivos do levantamento bibliográfico, que deram base e referências para o estudo das técnicas e delimitação dos softwares, e processos da análise, além de entender como age e funciona um malware.

Por fim, concluo dizendo que está é uma análise menos profunda, um pouco mais simples, para se entender e demonstrar os princípios de uma análise de *malware*. Um próximo passo, para um analisar experiente, seria utilizar-se de disassemblers e debuggers para análise do código em assembly do *malware*, executando e analisando linha por linha de seu código para todo seu entendimento.

Análise de *malware* é um processo que se necessita paciência, segurança, conhecimento, dedicação, sendo fundamental para a segurança da informação.

ANALYSIS OF STATIC AND DYNAMIC TECHNIQUES FOR DETECTION OF MALWARE. André Luís Sanches Benini.

ABSTRACT

In the area of information security, many threats exist, being the biggest of which, in terms of quantity and danger, the malware. This term used for malicious code in general, which the objectives is, destroy, alter, spy, overload, among other functions. Therefore, it is essential to detect the malware, to avoid losses. In this way, the detection performed through analysis, a static, which it is not executed the object of analysis, and another dynamic, which performs during of its execution. For this reason, this research intends to perform and demonstrate how these analyzes work and are

so important for security information, identifying and demonstrating their processes, always seeking to acquire and pass information and knowledge, another vision, seeking growth and contribution to security information. Thereby applied methodology was due to the bibliography survey, performing static and dynamic analysis in a malware in a secure virtual environment created, extracting, understanding, discussing, the results achieved through this research and analyzes. This way, observed all the process is complex, which requires patience in a malware analysis, being observed that, method the static ends up being a bit less deep than the dynamic, however, for a complete analysis, a combination of the two to better understanding running, threats, and how to combat malware.

Keywords: Malware. Information security. Static Analysis. Dynamic Analysis.

REFERÊNCIAS

ALEXANDRIA, João Carlos Soares de. **Gestão da Segurança da Informação – Uma Proposta para Potencializar a Efetividade da Segurança da Informação em Ambiente de Pesquisa Cientifica.** São Paulo: 2012. Disponível em:<http://www.iaea.org/inis/collection/NCLCollectionStore/_Public/43/130/43130675.pdf>. Acesso em: 19 abr. 2017.

COZZOLINO, Marcelo Freire. **Detecção de Variantes Metamórficas de Malware por Comparação de Códigos Normalizados**. Brasília: 2012. Disponível em: <http://repositorio.unb.br/bitstream/10482/10421/3/2012_MarceloFreireCozzolino.pdf>. Acesso em: 09 maio 2017.

FERNANDES, Jorge Henrique Cabral. **Gestão da segurança da informação e comunicações**. 1 ed. Brasília: Faculdade de Ciência da Informação, 2010.

IDIKA, Nwokedi; MATHUR, Aditya. **A survey of malware detection techniques.** West Lafayette: 2007 Disponível em: <https://www.researchgate.net/publication/229008321_A_survey_of_malware_detection_techniques>. Acesso em: 10 abr. 2017.

MCCLURE, Stuart; SCAMBRAY, Joel; KURTZ, George. **Hackers expostos**: **Segredo e Soluções para a Segurança de Redes**. 7 ed. Porto Alegre: Bookman, 2014.

MELO, L. P. D. et al. **Análise de Malware: Investigação de Códigos Maliciosos Através de uma Abordagem Prática.** Brasília: 2010 Disponível em:

<http://www.peotta.com/sbseg2011/resources/downloads/minicursos/90650.pdf>. Acesso em: 19 abr. 2017.

PINHEIRO, José Maurício Dos Santos. **Ameaças e Ataques aos Sistemas de Informação: Prevenir e Antecipar.** 2007. Disponível em: <http://web.unifoa.edu.br/cadernos/edicao/05/11.pdf>. Acesso em: 19 abr. 2017.

SIKORSKI, Michael; HONIG, Andrew. **Practical malware analysis: The Hands-On Guide to Dissecting Malicious Software**. 1 ed. San Francisco: No Starch Press, 2012.

TEWARI, Ayush; RASTOGI, Ishan; AGARWAL, Shubanshu. **Malware Detection Techiniques.** Índia: 2013. Disponível em: < https://web.iiit.ac.in/~ayush.tewari/RIS.pdf > Acesso em: 17 maio 2017.