Agentes de Análise e Detecção de Ataques Ransomware em Sistemas Linux

Cryslene Coêlho De Oliveira Miranda[1], Caio Carvalho Moreira[1], Otávio Noura Teixeira[1]

[1]Faculdade de Engenharia de Computação – Universidade Federal do Pará
68455-747 – Tucuruí – PA – Brasil

{cryslenecl, caioxmoreira, onoura}@gmail.com

Abstract. *Ransomware is a malware that affects the data security of systems or devices. It blocks user files by encrypting them and it requires a ransom in exchange for the decryption key. This work presents the development of a solution that aims to detect this threat, in Linux systems, through static agents. The algorithms are built in Python, aided by three programs: inotify, file and strace. Validation tests had been performed, in the perspective of presenting the efficiency of these agents.*

Resumo. *Ransomware é um malware que afeta a segurança de dados de sistemas ou dispositivos. Ele bloqueia os arquivos de usuários criptografando-os e exige um resgate em troca da chave de descriptografia. Este trabalho apresenta o desenvolvimento de uma solução que visa detectar essa ameaça, em sistemas Linux, por meio de agentes estáticos. São utilizados algoritmos feitos em Python, auxiliados por três programas: inotify, file e strace. Foram realizados testes de validação, na perspectiva de apresentar a eficiência desses agentes.*

1. Introdução

Segundo uma publicação[2] feita pela Kaspersky (empresa de antivírus e especialista em segurança digital), em seu site oficial, os ataques causados pelo *Ransomware* aumentaram significativamente contra diversas empresas no período de pandemia da COVID-19, já que a maioria das pessoas passou a trabalhar *home office* (escritório em casa). Eugene Kaspersky (cofundador e CEO da Kaspersky) explica que sem a proteção necessária, que deve ser oferecida pelas empresas, os funcionários se tornaram alvos fáceis de cibercriminosos. De acordo com os dados obtidos por uma pesquisa realizada por Dmitry Bestuzhev (diretor da equipe global de pesquisa e análise da Kaspersky na América Latina), o Brasil está à frente dos países que mais sofreram de ataques de *Ransomware* em empresas no primeiro semestre de 2020, como mostra a Figura 1 [3].

[2] Disponível em <https://www.kaspersky.com.br/blog/ransomware-telecommuting/15524/>

[3] Disponível em <https://www.kaspersky.com.br/blog/empresa-brasil-ransomware-pandemia/15527/>

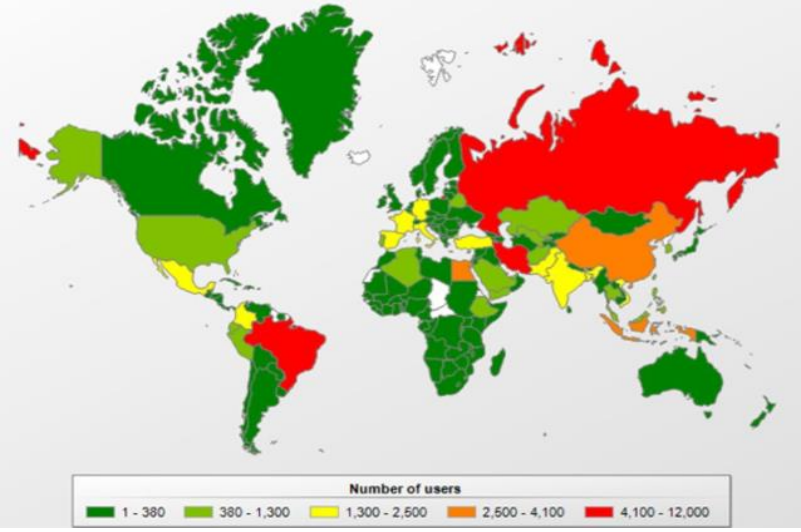

Figura 1. Países que mais sofreram de ataques de *Ransomware* no primeiro semestre de 2020, direcionado a usuários corporativos. Fonte: Kaspersky[3].

As seções que organizam este artigo tratam de conceitos importantes partindo da problemática de que muitos usuários vêm sendo afetados pela disseminação desse *malware*, que cresce a cada dia em vários setores (economia, dados, informações etc.), causando sérios danos, como perdas de dados ou financeiros. Esses ataques podem gerar prejuízos imprevisíveis e até mesmo irreparáveis, por isso é indispensável a busca por métodos para detecção desse *software* malicioso. Assim, serão abordadas as características para a identificação de *Ransomware*, o desenvolvimento de dois agentes de detecção e os resultados obtidos através dos experimentos utilizados na tentativa de auxiliar os usuários contra a invasão desses ataques e a evitar perdas maiores.

O objetivo deste estudo é a criação de agentes responsáveis pela análise e detecção de ataques *Ransomwares* em sistemas Linux, com o auxílio de três programas – *inotify, file* e *strace*. Na parte final deste estudo, para demonstrar a eficácia dos agentes criados, são apresentados testes de validação realizados com o intuito de eliminar possíveis falsos positivos.

Na seção 2 são descritos trabalhos relacionados ao processo de detecção do *Ransomware*. Na seção 3 são apresentados os fundamentos conceituais e a descrição dos três programas que ajudaram na detecção de um ataque *Ransomware*. Na seção 4 são abordados os procedimentos metodológicos, detalhando o processo de monitoração desse *malware* e de validação para eliminar possíveis falsos positivos, através dos agentes, além dos resultados dos testes na seção 5.

2. Trabalhos relacionados

Análise e detecção desse *software* malicioso, tema do estudo, também foram foco de outros trabalhos. Em Scaife et al. (2016), é apresentado um sistema de detecção e alerta de atividades suspeitas em arquivos. O sistema denominado CryptoDrop utiliza um conjunto de técnicas e indicadores que, combinados, puderam indicar ação desse *software* malicioso e efetuar seu bloqueio. Para a detecção foram analisados (a) mudança de tipos de arquivos, (b) medição de similaridade de arquivos, (c) entropia de Shannon e (d) criação e deleção de arquivos. Os resultados mostraram a possibilidade de reduzir significativamente a quantidade de perda de dados da vítima. De quase 5100

(cinco mil e cem) arquivos disponíveis, uma análise experimental obteve uma perda mediana de apenas 10 (dez) arquivos.

Já Bahrani e Bidgly (2019) criaram um método de identificação de *Ransomware* utilizando mineração de processos para analisar e monitorar os processos dos logs de eventos disponíveis para obter informações de padrões desses ataques e algoritmos de classificação para classificar esses tipos de *malwares* através da extração dessas informações. Foram analisados 21 tipos de *Ransomware* e outras amostras benignas e tiveram como resultado a precisão de 95% para a detecção desse *malware*, com o êxito e precisão dos algoritmos "j48" e "floresta aleatória" para classificação dos tipos de *Ransomware*.

Para impedir perdas antes de detectar a ação do *Ransomware*, Feng et al. (2017) criaram arquivos como armadilhas para comportamentos maliciosos, fazendo com que o *malware* os encontre e opere, enquanto a criptografia é monitorada através de um módulo de monitor. Se algum processo mostrar uma ação maliciosa, ele será encerrado e será informado ao usuário através de um aviso exibido na tela. Para os testes iniciais, utilizaram o *Locky* como *Ransomware* para a criptografia e, mesmo com alguns atrasos de tempo, foi possível detectar a criptografia e interromper a operação do *malware* em um tempo hábil, com um baixo custo de recursos do sistema e sem perdas para o usuário.

Este trabalho possibilita a exploração de outras formas no tratamento do tema, no intuito de encontrar resultados semelhantes, ou até melhores, a outros trabalhos e colaborar com o avanço no combate aos *Ransomwares*.

3. Fundamentação Teórica

3.1. *Ransomware*

Ransomware é um *malware* que tem como objetivo infectar o sistema, criptografando e/ou bloqueando dados, informações ou arquivos do usuário, solicitando um resgate, fazendo com que o usuário venha a ressarcir um valor específico para recuperar os arquivos [Scaife et al. 2016]. Geralmente o método de pagamento solicitado se dá por meio de criptomoeda (moeda digital) ou por serviços pré-pagos, como por exemplo o Ukash [Liska and Gallo 2017], alguns *Ransomwares* possuem temporizadores para mostrar o prazo em que o pagamento deve ser feito, caso o pagamento não seja realizado o usuário pode perder seus arquivos [Gonzalez and Hayajneh 2017].

O primeiro *Ransomware* foi desenvolvido por Joseph Popp em 1989, chamado AIDS [Giri et al. 2006], e tinha como finalidade infectar o sistema através de disquete substituindo o arquivo AUTOEXEC.BAT, executando a criptografia dos arquivos depois de aguardar 90 reinicializações da máquina [Gonzalez and Hayajneh 2017].

Esse *malware* utiliza criptografia forte e pode fazer com que o usuário sofra perda de acesso ao sistema, perda de confidencialidade e vazamento de informações [Young and Yung 1996], entretanto, este *malware* é previsível, já que a construção do seu comportamento é bem definida [Nieuwenhuizen 2017].

De acordo com Liska e Gallo (2017), um ataque *Ransomware* possui cinco fases de execução, como mostra a Figura 2.

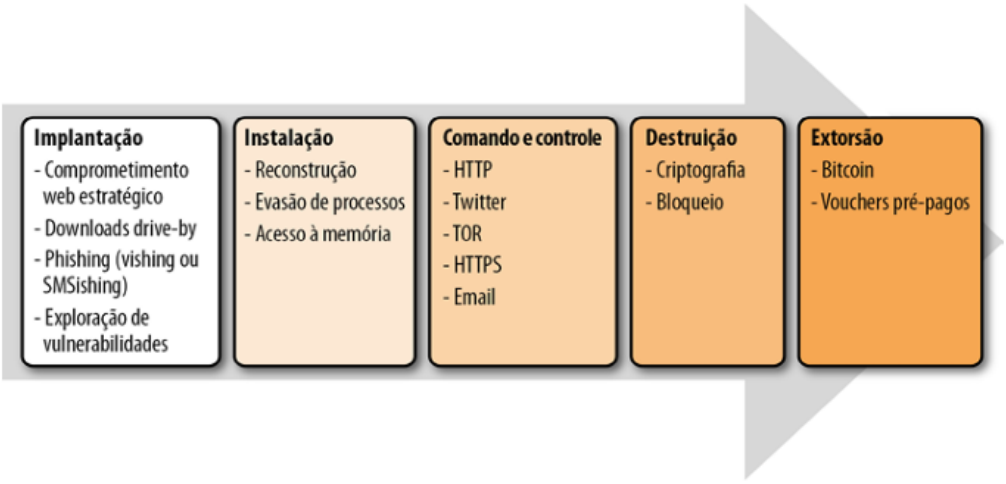

Figura 2. Anatomia de um ataque de *Ransomware* [Liska and Gallo 2017].

O foco deste trabalho se dá na quarta fase: destruição. Nesta etapa, os arquivos de utilização mais comuns pelos usuários começam a ser criptografados. Esses arquivos, geralmente, são reconhecidos através das extensões como "*.doc*", "*.txt*", "*.jpg*", "*.mp3*", "*.mp4*", "*.gif*", entre outros. Mesmo em sistemas Linux, onde a extensão não determina o tipo de arquivo, é usual que contenha a extensão para facilitar sua identificação.

Após o processo de criptografia, os dados úteis contidos nos arquivos ficam indisponíveis ao usuário, ou seja, estão bloqueados. Alguns tipos de *Ransomware* modificam o nome do arquivo para dificultar sua busca, por exemplo, de "*.doc*" o arquivo passa a ser "*doc.crypted*".

Cada tipo desse *malware* possui sua própria característica, com diferentes métodos de extorsão [Gonzalez and Hayajneh 2017]. Liska e Gallo (2017) descrevem com detalhes os tipos mais conhecidos e suas abordagens e peculiaridades.

Como o comportamento desse *software* malicioso é um dos aspectos mais importantes para sua análise, Nieuwenhuizen (2017) os classificou da seguinte forma:

- Classe A: Abre o arquivo e escreve sobre o arquivo infectado com a criptografia.
- Classe B: Move o arquivo para outro local, realiza a criptografia (de acordo com a classe A) e, por fim, devolve o arquivo para o diretório original.
- Classe C: Faz a leitura do arquivo, criptografando o conteúdo, depois salva o conteúdo que foi criptografado em um novo arquivo e exclui o arquivo original.

3.2. *Inotify*

Segundo Percivalle e Vanderlind (2015), o *inotify* é uma API do *kernel* do Linux e tem como objetivo principal fazer a monitoração dos eventos, tanto de arquivos quanto de diretórios, no sistema operacional, podendo notificar sobre as alterações de processos como criação, modificação etc.

O *inotify* pode ser utilizado como uma detecção de análise baseada em comportamento (dinâmica). A detecção baseada em comportamento é caracterizada pelo monitoramento dos processos em tempo real, para detectar se o comportamento é malicioso; caso seja, o processo pode alertar o sistema operacional, sendo eficaz pelo fato de que um ataque baseado em criptografia não muda de tipo para tipo de

Ransomware, pois contém características de comportamentos importantes para esse tipo de ataque [Nieuwenhuizen 2017].

3.3. *File*

File é um programa nativo do Linux que, pela leitura do cabeçalho do arquivo, consegue determinar qual é seu tipo. Este comando pode ser útil para análise de arquivos possivelmente alterados. Por exemplo, um arquivo de texto "*.txt*" é identificado como tipo "*ASCII text*". Porém, se o tipo de arquivo não for identificado, o *file* retorna o tipo "*data*", sendo esta a forma que um arquivo após ser criptografado se apresenta, significando que o mesmo não pode mais ser acessado pelo usuário.

3.4. *Strace*

Strace é um comando do terminal Linux que tem a finalidade de detectar registros de chamadas do sistema, sinais do sistema de rastreamento e erros de interceptação, que são chamados por processos e sinais determinados. O *strace* é um mecanismo ideal para análise e filtragem [Rincón Ques 2017]. Através de um arquivo de um executável é possível estudar o comportamento do processo e fazer a interferência nos eventos, podendo ser utilizado por algum algoritmo de detecção de ameaça [Teixeira et al. 2007].

4. Procedimentos Metodológicos

4.1. Monitoração

Para este projeto, foram utilizados os programas *inotify, file* e s*trace*, para identificar e detectar os padrões do *Ransomware* definidos nos algoritmos projetados, onde foi utilizada a linguagem de programação Python. Esses algoritmos são caracterizados como agentes que, segundo Russel e Norvig (2004), são "capazes de perceber seu ambiente por meio de sensores e de agir sobre esse ambiente por intermédio de atuadores". A Figura 3 ilustra as interações entre agente e ambiente.

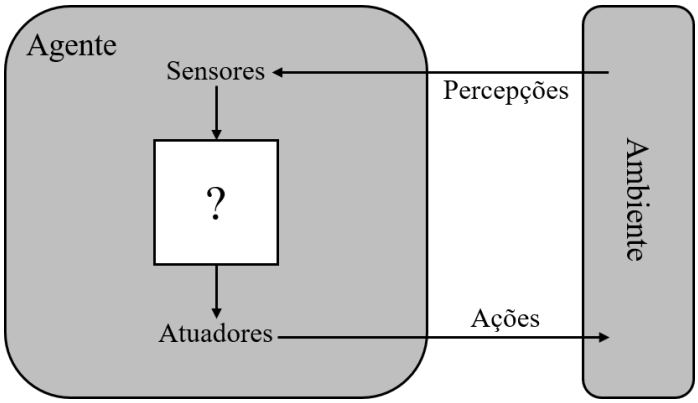

Figura 3. Agentes interagem com ambientes por meio de sensores e atuadores [RUSSEL and NORVIG 2004].

Para especificar o ambiente de tarefas, existe um modelo de descrição PEAS (*Performance*, *Environment*, *Actuator*, *Sensors* – desempenho, ambiente, atuadores e sensores) com os dados especificados para análise e detecção de ataque desse *software* malicioso [RUSSEL and NORVIG 2004], como mostra a Tabela 1.

Tabela 1. Descrição de PEAS para detecção de ataque *Ransomware*. [Adaptado de RUSSEL and NORVIG 2004]

Tipo de Agente	Medida de Desempenho	Ambiente	Atuadores	Sensores
Utilizando o *inotify*	Padrões de ataque *Ransomware*	Sistema Linux	Alerta de detecção	Eventos: *ACCESS* e *MODIFY*
Utilizando o *file*				Tipo de dado: *data*
Utilizando o *strace*				Eventos: *read*, *write* e *lseek*

O sistema operacional utilizado para os testes foi o *Kali GNU/Linux Rolling 64-bits*, executado em uma Máquina Virtual no Oracle VM VirtualBox com processador de 2 núcleos, 2GB de memória RAM, 20GB de memória secundária e 16MB de memória de vídeo.

A máquina física utilizada para execução da máquina virtual foi composta de um processador Intel Core i7-5500U CPU @ 2.4 GHz, 8GB de memória RAM, 1TB de memória secundária, executando Windows 10 Single Language 64-bits.

Foram utilizados para testes duas amostras de *Ransomware,* ambas desenvolvidas com fins educativos: 1) oriundo de um curso on-line[4] (*Udemy*) sobre o tema, implementado com Python e Pycrypto, e 2) *GonnaCry*[5] (*GitHub*), desenvolvido em *C*, que é similar ao conhecido *WannaCry*, porém, para Linux. Os dois *Ransomwares* podem ser considerados de classe A. Em ambos, a partir do momento em que o código é executado, o *malware* inicia o processo de criptografia dos arquivos que contém as extensões que foram definidas no código, como "*.doc*", "*.txt*", "*.jpg*", entre outras. O *GonnaCry*, além de executar a criptografia, altera o nome dos arquivos.

Para a realização dos testes, foram utilizados os arquivos disponibilizados para *download* gratuitamente pela Corpora Digital, que tem como proposta disponibilizar arquivos para uso em pesquisa e teste de ferramentas, esses arquivos incorporam diversos tipos de formatos de arquivos para testes em geral, simulando um conjunto normal de arquivos de um usuário comum [Garfinkel et al. 2009]. Foram escolhidos 5 (cinco) diretórios do conjunto de diretórios "*threads*" (/corpora/files/govdocs1/threads[6]), cada um com cerca de 1000 arquivos e 440MB, totalizando 2,16GB de dados, que serviram para os testes de tempo de criptografia e descoberta de padrões de ataques desse *software* malicioso. No caso dos dois *Ransomwares* utilizados, a criptografia do escopo de arquivos definidos demorou em média 06:07 (min:s), a análise desse tempo auxiliou na identificação dos padrões desse ataque, para que o usuário não sofresse uma perda significativa no seu sistema. Não será definido dados sobre o número de arquivos

[4] Disponível em <https://www.udemy.com/course/criando-ransomwares-com-python/learn/lecture/14122005#overview>

[5] Disponível em <https://github.com/tarcisio-marinho/GonnaCry>

[6] Disponível em <http://downloads.digitalcorpora.org/corpora/files/govdocs1/threads/>

criptografados nos testes com os dois agentes, pois o *Ransomware* não criptografa pela quantidade de arquivos e sim pelo tamanho dos arquivos, por exemplo, se em uma pasta tem apenas 1 (um) arquivo de 1GB e em outra pasta tem 10 (dez) arquivos diversos que somam 1(GB), o *malware* irá criptografar as duas pastas com o mesmo período de tempo.

4.2. Monitoração com *inotify* e *file*

Para a detecção do *Ransomware,* foi criado um código em Python como agente para monitoração de alterações de arquivos e/ou diretórios, com auxílio dos programas *inotify* e *file*.

Utilizando o módulo "*PyInotify*", que tem acesso ao *inotify*, conseguiu-se a monitoração em tempo real de qualquer alteração feita nos arquivos e/ou diretórios. Como esse *malware* faz acessos e modificações em números significativamente altos de arquivos, o foco para sua identificação foi baseado na quantidade de eventos *ACCESS* e *MODIFY*. Porém, se esses eventos forem analisados isoladamente, a taxa de falso positivo pode ser alta.

O *file* foi utilizado para confirmar a detecção da criptografia. Ele por sua vez identifica o tipo de dado dos arquivos que estão sendo modificados, se o tipo de dado do arquivo estiver com a descrição "*data*" ao invés da definição do seu tipo de dado usual, significa que o arquivo foi modificado pelo *malware*.

A quantidade de alterações feitas pelo *Ransomware* foi determinada com o valor de aproximadamente um terço (1/3) da média da quantidade de *ACCESS* e *MODIFY* (para o *inotify*) e da quantidade total de tipos de dados não identificados "*data*" (para o *file*) de acordo com o tempo de criptografia, como mostra a Equação (1), sendo que esse parâmetro foi definido de forma empírica, envolvendo análise do tempo de criptografia, tamanho dos diretórios e a quantidade de eventos, onde foi acompanhado todo o processo durante a execução, obtendo as informações importantes do comportamento da criptografia, para que o usuário fosse alertado a tempo de se proteger de perdas maiores.

$$\begin{aligned} Qtd_{ACC} &\cong (Média_{ACC}) * 1/3 \\ Qtd_{MOD} &\cong (Media_{MOD}) * 1/3 \\ Qtd_{TDA} &\cong (Media_{TDA}) * 1/3 \end{aligned} \quad (1)$$

Legenda:

Qtd_{ACC} – Valor para a quantidade de *ACCESS*

Qtd_{MOD} – Valor para a quantidade de *MODIFY*

Qtd_{TDA} – Valor para a quantidade de tipos de dados alterados

$Média_{ACC}$ – Média das quantidades das 5 (cinco) execuções de *ACCESS*

$Media_{MOD}$ – Média das quantidades das 5 (cinco) execuções de *MODIFY*

$Media_{TDA}$ – Média das quantidades das 5 (cinco) execuções de tipos de dados alterados

Assim, com o auxílio de um contador, foi definido que se (e somente se), a partir do *inotify* houver mais de 25000 (vinte e cinco mil) *ACCESS* e mais de 50000 (cinquenta mil) *MODIFY* e a partir do *file* tiver mais de 500 (quinhentos) arquivos

modificados para o tipo de dado "*data*", o sistema para o *loop* de monitoração e alerta que o *Ransomware* foi detectado.

A Tabela 2 mostra as 5 (cinco) execuções para analisar a quantidade total de *ACCESS* e *MODIFY* e alterações do tipo de dados quando o *Ransomware* está atacando.

Tabela 2. Dados obtidos através do *inotify* e *file* (Elaborado pelo autor)

Nome do diretório	Quantidade *ACCESS*	Quantidade *MODIFY*	Tipos de dados alterados
thread1	84334	162991	1627
thread2	79218	154498	1608
thread3	78086	151549	1608
thread4	82894	162926	1605
thread5	80991	159400	1620

4.3. Monitoração com *strace*

Para a detecção do *Ransomware* utilizando o *strace*, foi criado um algoritmo em Python como agente que analisa as chamadas de sistemas feitas durante a criptografia, com o auxílio das bibliotecas "*numpy*", "*os*" e "*csv"*. Através da biblioteca "*os*", foi possível usar o comando *strace* com o parâmetro "-o" para salvar em uma planilha (*.csv*) o resumo das informações das chamadas do sistema dado pelo parâmetro "-c". Com as bibliotecas "*numpy*" e "*csv*" foi possível ler esse arquivo salvo e definir os padrões analisados para detectar a atuação desses ataques.

O comando "*strace -c comando*" tem seu resultado em forma de uma tabela com suas informações específicas, como mostra a Tabela 3.

Tabela 3. Exemplo "strace -c comando" (Adaptado da tela impressa no terminal do Linux)

% time	seconds	usecs/calls	calls	errors	syscall
----------	----------	----------	----------	----------	----------
100.00					total

A tabela salva em formato "*.csv*" foi transformada para matriz, já que originalmente ela é salva em forma de lista. De todos esses dados, foram analisadas somente as colunas "*seconds*", "*calls*" e "*syscall*" para definir o padrão de um ataque desse *software* malicioso.

O *Ransomware* faz chamadas de leitura (*read*) e escrita (*write*) em um número consideravelmente alto e em um tempo de processamento (*seconds*) muito curto, além disso ele reposiciona o deslocamento (*lseek*) para cada arquivo de leitura e escrita, por isso a quantidade de chamadas para *lseek* é bem maior do que de leitura e escrita.

A quantidade de alterações feitas por esse *malware* foi determinada com o valor de aproximadamente um terço (1/3) da média da quantidade de *read, write* e *lseek* de acordo com o tempo de criptografia, como mostra Equação (1), o contador utilizado é da própria configuração do *strace* que contabiliza todas as chamadas realizadas e mostra

para o usuário. Assim, foi definido que se (e somente se) houver 100000 (cem mil) *read* e 200000 (duzentos mil) *write* e 300000 (trezentos mil) *lseek*, ele alerta que teve atuação de *Ransomware* no sistema, sendo esse o padrão identificado para as chamadas de sistemas quando esse *software* malicioso faz alterações nos arquivos e diretórios.

A Tabela 4 mostra os dados de 5 (cinco) testes para definição dos valores para a identificação da criptografia, de acordo com quantidade total de dos eventos e o tempo de ação da criptografia.

Tabela 4. Dados obtidos através do *strace* (Elaborado pelo autor)

Nome do diretório	Quantidade *read*	Quantidade *write*	Quantidade *lseek*
thread1	120900	238171	361889
thread2	120188	241215	362486
thread3	118268	237554	361548
thread4	127966	242654	364605
thread5	124155	241525	363332

Diferente do algoritmo que utiliza o *inotify* e *file* que para a monitoração assim que detecta um ataque, o resultado da detecção com o *strace* só é mostrado ao final da execução do processo. Portanto, o agente criado com o auxílio do *strace* é um complemento para a confirmação da ação presente no *Ransomware* no sistema Linux.

4.4. Diminuindo falsos positivos

É possível que determinados programas de usuário, tendo ações similares ao de um *Ransomware*, sejam confundidos com este. Este resultado seria classificado como falso positivo. Para diminuir a incidência de possíveis falsos positivos, foram testados dois programas com ações de alteração em múltiplos arquivos, comportamento característicos do *malware*: 1) *rename* e 2) *Adobe Photoshop*.

Com o programa *rename*, os nomes dos arquivos e diretórios foram convertidos de letras minúsculas para letras maiúsculas. Já com o Adobe *Photoshop*, foram editadas as imagens dos diretórios especificados, acrescentando marca d'água em múltiplas fotos em sequência, esse processo é similar a ação de um *Ransomware* que, ao criptografar, altera o conteúdo de diversos arquivos em sequência.

No código para a detecção do *malware* usando o *inotify*, o contador incrementa quando detecta a string definida ("*IN_ACCESS*" e "*IN_MODIFY*"). As alterações executadas manualmente, através de programas ou pelo próprio sistema operacional, têm eventos a mais que de um ataque *Ransomware* e a quantidade de *ACCESS* e *MODIFY* são bem menores. Além disso, o *file* confirma que a alteração foi realizada, pois mostra que o conteúdo do arquivo não é mais possível de ser acessado.

O algoritmo utilizando o *strace* também foi testado com o programa *rename*, onde ele mostra as chamadas de sistemas quando os arquivos especificados foram renomeados ao mesmo tempo.

Também foi realizado o teste de validação para o programa *Photoshop,* com o agente utilizando o *strace*, acrescentando marca d'água em diversos arquivos ao mesmo tempo. A diferença entre a detecção da criptografia e a atuação do *Photoshop* é a

quantidade de eventos e tipos de chamadas do sistema, para eliminar a possibilidade de apresentar possíveis falsos positivos.

5. Resultados

5.1. Detecção a partir do *inotify* e *file*

Assim que o algoritmo que utiliza o *inotify* e o *file* detecta o *malware* a partir dos padrões definidos para detecção, o usuário é alertado de que está sofrendo um ataque desse *software* malicioso, como mostra a Figura 4. Para que a modificação de arquivos seja considerada um ataque de *Ransomware*, o *inotify* e o *file* precisam satisfazer os padrões identificados.

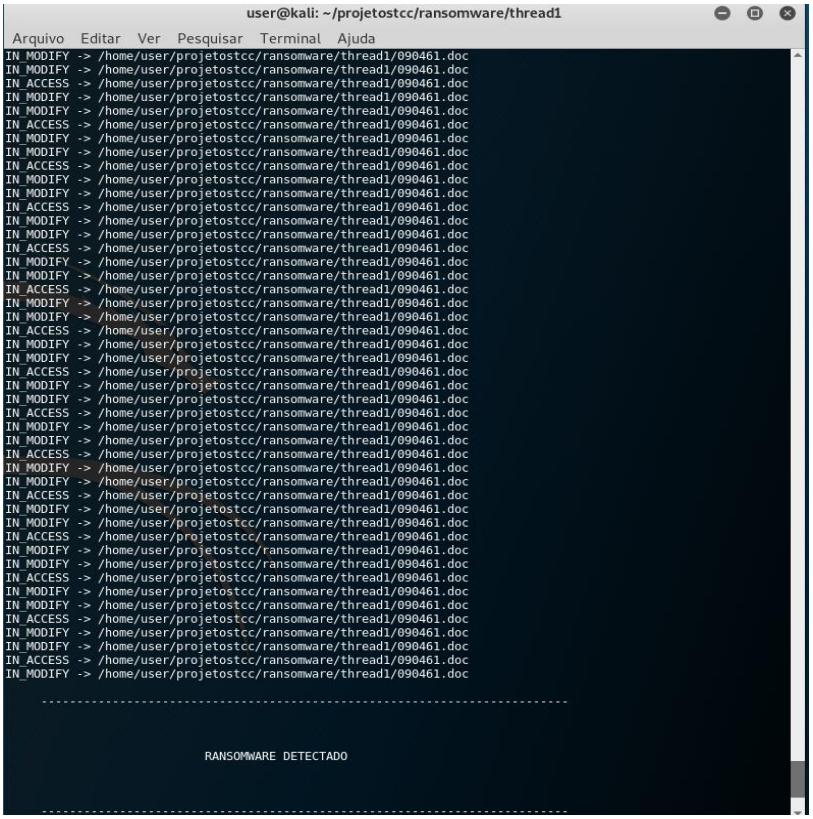

Figura 4. Identificação do *Ransomware* através do *inotify* e *file* (Elaborado pelo autor)

Dentre os 5 (cinco) testes realizados como o *rename*, nenhum detectou a presença do *Ransomware*, como mostra a Tabela 5. Os eventos *ACCESS* e *MODIFY* apresentaram mais caracteres do que foi definido para o agente de detecção e o conteúdo dos arquivos não foram alterados, eliminando a possibilidade de apresentar possíveis falsos positivos.

Tabela 5. Dados dos testes realizados com o programa *rename* (Elaborado pelo autor)

Nome do diretório	Quantidade *ACCESS*	Quantidade *MODIFY*	Tipos de dados alterados	Falso Positivo
thread1	930	0	0	Não
thread2	890	0	0	Não
thread3	890	0	0	Não
thread4	912	0	0	Não
thread5	895	0	0	Não

A Tabela 6 mostra dados de 5 (cinco) execuções do *Photoshop*, enfatizando a quantidade de *ACCESS*, *MODIFY* e os tipos de dados alterados, sendo que nenhum dos testes realizados detectaram uma falsa ação do *malware*. Algumas amostras dos eventos *ACCESS* e *MODIFY* que entram na contagem é menor do que foi definido para o agente de detecção, na maioria das vezes, além disso, apesar do conteúdo do arquivo ter sido alterado por ter acrescentado a marca d'água, nenhum de tipo de dado dos arquivos foram alterados, sendo possível de o usuário ter acesso a esses arquivos, eliminando a possibilidade de apresentar possíveis falsos positivos.

Tabela 6. Dados dos testes realizados com o programa *Photoshop* (Elaborado pelo autor)

Nome do diretório	Quantidade *ACCESS*	Quantidade *MODIFY*	Tipos de dados alterados	Falso Positivo
thread1	20289	45025	0	Não
thread2	20542	43523	0	Não
thread3	19348	40965	0	Não
thread4	25234	50159	0	Não
thread5	22851	45963	0	Não

Como pode ser observado na Tabela 6, no diretório "*thread4*" foram modificadas algumas imagens, com a incrementação da marca d'água realizada, sendo que esses acessos e modificações ultrapassaram o limite definido para os padrões de identificação de um *Ransomware* para o *inotify*, mas devido o auxílio do *file,* foi possível afirmar que não teve um ataque, já que continuou sendo possível o acesso ao conteúdo do tipo da imagem, eliminando assim uma possível afirmação de um falso positivo.

5.2. Detecção a partir do *strace*

A Figura 5 mostra a tela que aparece ao usuário quando algoritmo, que utiliza o *strace*, detecta a atuação de um ataque de *Ransomware* a partir dos padrões definidos.

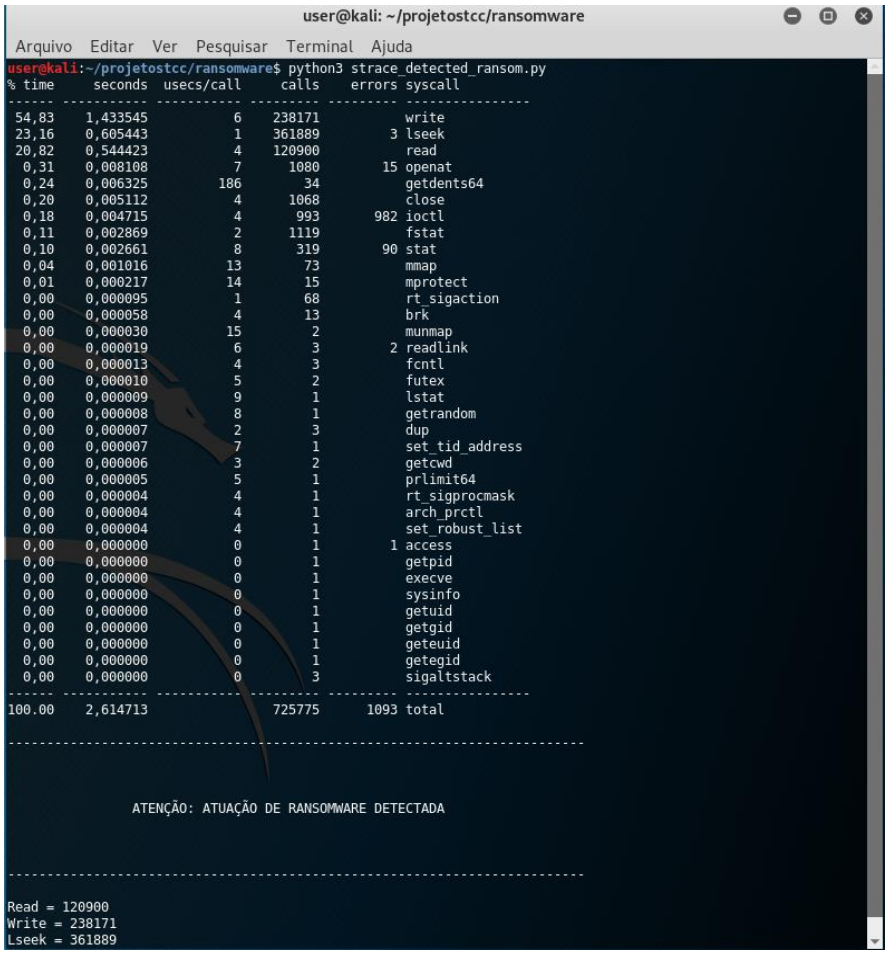

Figura 5. Identificação do *Ransomware* através do *strace* (Elaborado pelo autor)

Na Tabela 7 pode-se observar os dados dos 5 (cinco) testes realizados com o programa *rename*, enfatizando os valores do *write, read* e *lseek*; dentre os testes nenhum detectou a presença do *malware*.

Tabela 7. Dados dos testes realizados com o programa *rename* (Elaborado pelo autor)

Nome do diretório	*Quantidade read*	*Quantidade write*	*Quantidade lseek*	*Falso Positivo*
thread1	2965	0	6152	Não
thread2	2512	0	5962	Não
thread3	2348	0	5710	Não
thread4	2848	0	6103	Não
thread5	2741	0	6001	Não

A Tabela 8 mostra as 5 (cinco) execuções de testes utilizando o *Photoshop*, enfatizando a quantidade de *read*, *write* e *lseek*; desses testes em apenas 1 (um) foi detectada a presença do *software* malicioso.

Tabela 8. Dados dos testes realizados com o programa *Photoshop* (Elaborado pelo autor)

Nome do diretório	Quantidade *read*	Quantidade *write*	Quantidade *lseek*	Falso Positivo
thread1	94205	192241	295516	Não
thread2	91738	188353	291962	Não
thread3	89871	182841	288575	Não
thread4	95483	190894	294130	Não
thread5	**118653**	**207157**	**312294**	**Sim**

6. Conclusão

Ao confrontar os resultados aos objetivos propostos no decorrer deste estudo, ratificou-se a importância e a necessidade de analisar e propor um caminho para a detecção de ataque *Ransomware* em sistemas Linux.

Os códigos para os agentes mostraram ser eficientes na maioria dos cenários demonstrados. O *inotify* proporcionou o auxílio na monitoração da criptografia e com os padrões identificados foi possível obter a detecção nos testes realizados com os dois modelos de *Ransomware* apresentados. Com o *file* foi possível fazer com que a identificação do *malware* fosse mais precisa, onde o tipo de arquivo pode eliminar os possíveis falsos positivos que possam surgir. O *strace*, com a análise de chamadas de sistemas para a detecção do *software* malicioso*,* também se mostrou competente para a identificação dos padrões, em relação às alterações feitas manualmente ou pelos sistemas. Com isso, foi possível monitorar e identificar o ataque do *Ransomware* nos testes realizados e para uma análise mais confiável, foi necessário realizar testes para validação na perspectiva de apresentar a eficiência dos códigos dos agentes, onde tentou-se diminuir a taxa de falsos positivos, pois na maioria dos testes realizados, através de alterações manuais ou do sistema, apenas uma das modificações realizadas foi confundida com um ataque de um *Ransomware*, a maioria mostrava eventos, dados de arquivos ou processos diferentes (tipos e quantidades) dos padrões identificados e definidos para detecção.

A abordagem feita com outros trabalhos mostra a diversidade de processos que podem ser utilizados para monitoração e detecção de um ataque de *Ransomware*, como o sistema CryptoDrop, de Scaife et al. (2016), que identifica e bloqueia a ação desse *malware* para diminuir possíveis perdas dos usuários, o trabalho de Bahrani e Bidgly (2019) que, através de mineração de processos e algoritmos de seleção, tiveram êxito ao identificar a ação da criptografia e classificar os tipos detectados e o projeto de Feng et al. (2017), onde criaram arquivos como artifícios para atrair o ataque de um *Ransomware* e analisar o comportamento através de suas características. Assim, o método deste projeto cumpre a finalidade de monitorar e detectar um ataque de *Ransomware* utilizando dois agentes estáticos, além de abordar um tema preocupante e que está em constante evolução.

Em projetos futuros pode-se incrementar os agentes propostos, analisando os padrões para outros tipos de *Ransomware* ou implementar métodos de bloqueio contra ataques realizados por esses *softwares* maliciosos, no qual podem ser auxiliados pelos

programas *inotify*, *file* e *strace* para realizar *backups* de todo sistema ou bloquear esses ataques, por exemplo, além de poder implementar agentes inteligentes e/ou bioinspirados a fim de se adotar processos cada vez mais eficazes contra esses ataques. Como o *Ransomware* é um *malware* que está em constante adaptação e melhorias de sofisticação, também pode ser necessário realizar algumas modificações nos agentes, de acordo com os padrões identificados.

Referências

Bahrani, A. and Bidgly, A. J. (2019). *Ransomware detection using process mining and classification algorithms. In2019 16th International ISC (Iranian Society of Cryptology) Conference on Information Security and Cryptology (ISCISC)*, pages 73–77.

Feng, Y., Liu, C., and Liu, B. (2017). Poster: A new approach to detecting ransomware with deception. In 38th IEEE Symposium on Security and Privacy.

Garfinkel, S., Farrell, P., Roussev, V., and Dinolt, G. (2009). Bringing science to digital forensics with standardized forensic corpora. digital investigation, 6:S2–S11.

Giri, B. N., Jyoti, N., and Avert, M. (2006). The emergence of ransomware. AVAR, Auckland.

Gonzalez, D. and Hayajneh, T. (2017). Detection and prevention of crypto-ransomware. In *2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON)*, pages 472–478. IEEE.

Liska, A. and Gallo, T. (2017). Ransomware: defendendo-se da extorsão digital. *São Paulo: Editora Novatec.*

Nieuwenhuizen, D. (2017). A behavioural-based approach to ransomware detection. *Whitepaper. MWR Labs Whitepaper.*

Percivalle, D. and Vanderlind, S. (2015). Oversubscribing inotify on embedded platforms. California, EUA. California Polytechnic State University San Luis Obispo.

Rincón Ques, P. (2017). Estudi eines de traça: ltrace, strace i dtrace. Barcelona, Espanha. Universitat Oberta de Catalunya.

RUSSEL, S. J. and NORVIG, P. (2004). Inteligência artificial: uma abordagem moderna. 2a edição. *Rio de Janeiro, Brasil. Editora Campus.*

Scaife, N., Carter, H., Traynor, P., and Butler, K. R. (2016). Cryptolock (and drop it): stopping ransomware attacks on user data. In *2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS)*, pages 303–312. IEEE.

Teixeira, E., Antunes, J., and Neves, N. (2007). Avaliação de ferramentas de análise estática de código para detecção de vulnerabilidades. *Faculdade de Ciências da Universidade de Lisboa, Departamento de Informática*, page 21.

Young, A. and Yung, M. (1996). Cryptovirology: Extortion-based security threats and countermeasures. In *Proceedings 1996 IEEE Symposium on Security and Privacy*, pages 129–140. IEEE.