# Análise de Amostra de *Ransomware* WannaCry

Rodrigo P. O. F. Braga, Guilherme P. Aquino e Evandro C. Vilas Boas

***Resumo*— Esse trabalho examina uma amostra do *ransomware* WannaCry usando análise estática e dinâmica para compreender seu comportamento durante um ataque. Discutem-se os objetivos de cada análise e apresenta-se a configuração do ambiente controlado para estudo. Na análise estática, utiliza-se a ferramenta de análise de binários executáveis PeStudio para verificar arquivos executáveis, características da amostra, funções de criptografia e `imports`. Executa-se a amostra e utiliza-se as ferramentas Procmon e WireShark para coletar evidências de interações do *ransomware* com o ambiente. Os resultados da análise estática são discutidos e comparados com evidências da análise dinâmica, permitindo compreender o comportamento do *ransomware*.**

***Palavras-Chave*— Análise, *malware*, *ransomware*, segurança cibernética, WannaCry.**

***Abstract*— This work evaluates a WannaCry ransomware sample using static and dynamic analyses to comprehend its behavior during an attack. It delves into the objectives of each analysis and presents the controlled environment setup used for studying. In static analysis, the PeStudio executable binary analysis tool checks executable files, sample characteristics, encryption functions, and `imports`. The sample is executed, and Procmon and WireShark tools are utilized to gather evidence of the interactions between ransomware and the environment. The static analysis findings are then discussed and compared with evidence from the dynamic analysis, allowing for an understanding of the ransomware behavior.**

***Keywords*— Analysis, cybersecurity, *malware*, *ransomware*,*WannaCry*.**

## I. Introdução

*Softwares* maliciosos (*malicious softwares*, *malwares*) são criados por diversas razões, como para obter acesso não autorizado à informações visando lucro financeiro ou para causar dano à integridade de sistemas [1]. A disseminação de *malwares* ocorre de várias maneiras, incluindo exploração de vulnerabilidades em sistema e uso de engenharia social [2]. Ataques de *malwares* podem ser implementados por meio de campanhas de *phishing* entregues via e-mail e/ou por *download* e execução de *malware* pelo próprio usuário, acreditando se tratar de um *software* ou documento legítimo [1]. Invasores também podem explorar vulnerabilidades em sistema como vetores de entrada para inserir e executar um *malware*.

Dentre os *malwares* com maior potencial de dano econômico, destaca-se o *ransomware*, cujo objetivo é bloquear acesso aos sistemas e/ou arquivos de um usuário ou corporação através de criptografia [3]. Após executar tais funções, esse *malware* realiza o processo de extorsão financeira ao dispor uma mensagem de resgate ao usuário exigindo um pagamento para restaurar o acesso ao sistema ou arquivos [4]. A complexidade do *ransomware* esta relacionada ao uso de diversas técnicas para criptografar os dados de forma rápida e eficiente, se propagar pela rede sem ser identificado e infectar o máximo de sistemas possíveis [5].

Há basicamente dois tipos principais de *ransomware*: *crypto* e *locker* [6]. *Ransomwares* do tipo *crypto* realizam a criptografia de arquivos, tornando-os inacessíveis para o usuário. A recuperação do acesso aos arquivos é possível por meio da decriptografia com uso da chave de criptografia em posse do agente malicioso, autor ou detentor do *ransomware*. Já os *ransomwares* do tipo *locker* não criptografam os arquivos, mas realizam o bloqueio do sistema e exibem uma tela ao usuário para comunicar o ataque sempre que há tentativas de realizar qualquer função na máquina infectada. No caso do *crypto*, a extorsão financeira tem como motivação recuperar o acesso aos dados criptografados. Para os *ransomwares* do tipo *locker*, o pedido de resgate tem como premissa devolver o acesso do usuário ao sistema.

A sofisticação e frequência dos ataques de *ransomwares* os tornam uma ameaça significativa à qualquer negócio. Essa prática é endossada por modelos econômicos de negócios como o *Ransomware as a Service* (RaaS) que permite a fácil veiculação de kits *ransomware*, muitas vezes acompanhados de suporte técnico para facilitar a adesão de agentes maliciosos aos ataques dessa natureza [7]. Existe ainda a problemática do polimorfismo binário das amostras, uma técnica que permite alterar suas estruturas binárias de forma dinâmica, enquanto mantém sua funcionalidade principal [8], [9]. Isso é feito para alterar a assinatura da amostra e evitar detecção por soluções de segurança baseadas nessa característica.

Os danos causados por ataques de *ransomware* vão além dos valores em resgate pagos pelas vítimas. Há custos adicionais que incluem a recuperação dos dados, o lucro perdido pela interrupção dos negócios e danos à reputação. Os custos ocultos como remediação e perda de receita, podem exceder significativamente o valor do resgate. Portanto, torna-se imprescindível compreender a natureza de um *ransomware*, bem como os principais mecanismos de funcionamento associados a esse tipo de *malware*. Esse conhecimento permite desenvolver estratégias eficazes em mitigação e conscientização a respeito desse tipo de ameaça cibernética.

Esse trabalho examina uma amostra do *ransomware* WannaCry através de análise estática e dinâmica para avaliar a presença de mecanismos presentes em um ataque de *ransomware*. O *ransomware* WannaCry é um *cripto-ransomware*, veiculado em 2017, que afetou milhares de sistemas [10]. Esse *ransomware* explorou a vulnerabilidade conhecida como EternalBlue do sistema operacional Windows, que afeta o

Rodrigo P. O. F. Braga, Guilherme P. Aquino e Evandro C. Vilas Boas , Centro de Segurança Cibernética do Inatel (CxSC Telecom) e Cyber Security and Artificial Intelligence (CS&I Lab.), Instituto Nacional de Telecomunicações (Inatel), Santa Rita do Sapucaí - MG, 37540-000, Brasil, e-mail: rodrigo.paiva@gec.inatel.br, guilhermeaquino@inatel.br, evandro.cesar@inatel.br. Este trabalho foi parcialmente financiado pela Fundação de Amparo à Pesquisa do Estado de Minas Gerais (Fapemig), Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e Centro de Segurança Cibernética do Inatel (CxSC Telecom).

protocolo de compartilhamento de arquivos SMBv1 [11]. A amostra utilizada nesse trabalho foi retirada de um repositório público no GitHub[1], sendo a análise organizada em cinco seções. Na Seção II, discorre-se sobre a análise estática e dinâmica de um *ransomware*, bem como define-se a configuração do ambiente de testes. Os resultados obtidos por meio da análise estática são discutidos na Seção III, enquanto a análise dinâmica é apresentada na Seção IV. Na Seção V, abordam-se os principais comentários, conclusões e trabalhos futuros.

## II. Metodologia de análise

Essa seção discute os objetivos da análise estática e dinâmica na manipulação da amostra do *ransomware* WannaCry. Assim como, apresenta a configuração do ambiente controlado para execução e observação do comportamento do *malware*. O uso de um ambiente controlado e isolado evita danos à rede e ao dispositivo usado para análise.

### A. Análise estática e dinâmica de um ransomware

A análise de um *malware* pode ser dividida entre estática e dinâmica [12]. A análise estática investiga o código ou arquivo da amostra de *ransomware* sem antes executá-lo. Essa etapa tem como propósito identificar *strings* maliciosas, bibliotecas utilizadas, mecanismos de ofuscação e padrões de comportamento [13]. Ferramentas como PeStudio e PEID permitem obter os *footprints* e *hashes* dos componentes da amostra apenas analisando seu arquivo. Logo, é possível capturar DLL's e funções que a amostra utiliza para identificar a presença de diferentes módulos relacionados, por exemplo, à criptografia, à lateralização (contaminação de outros ambientes através de uma vulnerabilidade específica, um serviço de rede), à evasão de detecção (tentativa de burlar soluções em anti-vírus e *firewalls*), entre outros. Durante a análise estática levantam-se hipóteses iniciais que serão confirmadas ou não durante a análise dinâmica, sendo crucial para interpretar os resultados do experimento e guiar etapas futuras.

Na análise dinâmica, executa-se a amostra de *ransomware* em um ambiente controlado para observação de seu comportamento em tempo real. Diversas ferramentas podem ser usadas para validar as interações que a amostra tenta realizar com o ambiente. Ferramentas que detectam alterações em arquivos como *Filemon* ou *Procmon* permitem identificar arquivos criptografados. Ferramentas como o Wireshark são úteis para viabilizar o monitoramento de rede para posterior análise de pacotes enviados pela amostra durante a execução. Isso permite verificar se o *ransomware* faz contato com algum servidor externo, assim como coleta evidências de replicação da amostra pela rede [14].

A combinação de resultados entre a análise estática e dinâmica provê uma compreensão abrangente do *ransomware* quanto à estrutura, funcionalidades e potencial impacto ao sistema. Entretanto, o autor do *ransomware* pode dificultar a análise e compreensão do código ao implementar técnicas de empacotamento e ofuscação. O empacotamento consiste em comprimir ou criptografar o código do *ransomware* para dificultar a leitura do arquivo e protegê-lo contra mecanismos simples de detecção baseados apenas em padrões de código. A ofuscação modifica o código em si com funções e variáveis com nomes aleatórios e porções de código irrelevantes que tornam sua interpretação complexa. Em complemento, a amostra pode apresentar mecanismos ocultos de evasão e defesa. Por exemplo, o anti-*debugging* permite verificar o uso de ferramentas de *debug* e *flags* de depuração ou detecção de interrupções, enquanto o anti-VM detecta informações de sistema, *drivers* e virtualização, para que a amostra identifique execução em um ambiente virtual [15].

[1] https://github.com/Da2dalus/The-MALWARE-Repo

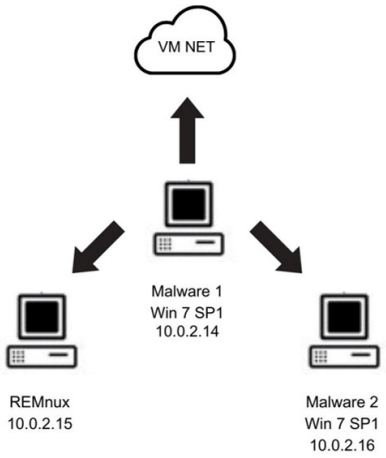


Fig. 1. Topologia da rede configurada para análise da amostra de *ransomware*.

### B. Configuração do ambiente controlado

O ambiente controlado foi implementado por meio da ferramenta de virtualização VirtualBox, em que foram criadas três VMs em topologia de rede, vista na Figura 1. Duas VMs foram configuradas com imagens do sistema operacional Windows 7 SP1 e a terceira contendo uma distribuição Linux chamada REMnux, especifica para a análise de *malwares*. Para evitar mecanismos de evasão e garantir maior responsividade durante os testes, todas as máquinas foram configuradas com as seguintes especificações técnicas: 3 CPU's, 2GB RAM e 50GB de armazenamento. As VMs foram configuradas em uma rede de acesso local e o ambiente foi propriamente isolado através das opções de compartilhamento de arquivos entre o programa de virtualização e a máquina.

A máquina "Malware1" é destinada a execução de todos os testes. Essa VM possui uma interface de rede configurada para a rede interna, servindo como interface de rede local à qual todas as outras máquinas são conectadas. As máquinas "Malware2" e "REMnux", serão utilizadas como parâmetro de teste para obter informações sobre tentativas de lateralização e outros mecanismos. A distribuição Linux oferece maior detalhe nas análises através de suas ferramentas embutidas, que serão discutidas propriamente durante a análise estática e dinâmica.

## III. Resultados da Análise Estática

A análise estática da amostra de *ransomware* WannaCry foi realizada utilizando a ferramenta PeStudio. PeStudio é uma

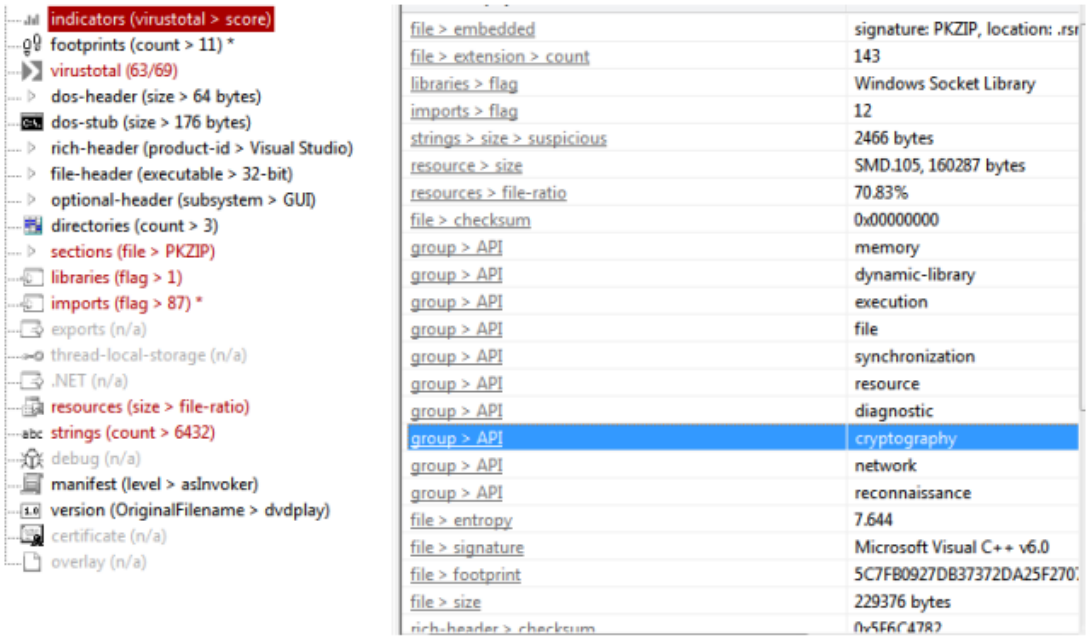


Fig. 2. Interface da ferramenta PeStudio demonstrando alguns resultados de análise estática para a amostra de *ransomware* WannaCry.

ferramenta de análise estática de binários executáveis para sistemas Windows. Essa ferramenta fornece uma visão detalhada dos arquivos executáveis e permite examinar e entender melhor o funcionamento de programas e bibliotecas dinâmicas (DLLs). A ferramenta permite avaliar *hashes* do *malware*, os `imports`, as DLLs afetadas, as funções utilizadas durante sua execução, entre outras informações relevantes.

Na análise do *ransomware*, o PeStudio retornou algumas informações iniciais a respeito da natureza do arquivo, identificando alguns módulos, conforme visto na Figura 2. Na aba "Indicator" da interface do PeStudio, tem-se informações a respeito dos módulos da amostra relacionados a criptografia, rede, reconhecimento, entre outros. O PeStudio possui um indicador denominado de "*entropy*" que pode assumir valores de 0 a 8, indicando a menor ou maior probabilidade do *malware* sob análise estar empacotado ou ofuscado [16]. Para a amostra sob análise, esse valor é de 7,64 indicando que o código malicioso está protegido, dificultando a análise e detecção por ferramentas de segurança. Para desempacotar adequadamente a amostra e analisar todos os seus recursos, é necessário conhecer o tipo de empacotador utilizado, como por exemplo UPX, themida ou DTPacker [17]. Em seguida, utilizar um programa auxiliar para reverter o estado da amostra e analisar seu código em baixo nível (Assembly). A aplicação dessa abordagem não garante a veracidade das informações analisadas, pois o autor do *ransomware* pode ocultar rastros do código durante a sua escrita. Por esse motivo, esses detalhes não serão abordados neste documento.

A ferramenta PeStudio também permite a análise das secções da amostra, permitindo verificar as permissões de execução (`read`/`write`) e a porcentagem que ocupam no arquivo fonte. Na Figura 3, verifica-se que 71,43% da amostra escolhida consistem em recursos como bibliotecas e *imports*, utilizadas pelo *ransomware* durante a rotina de execução.

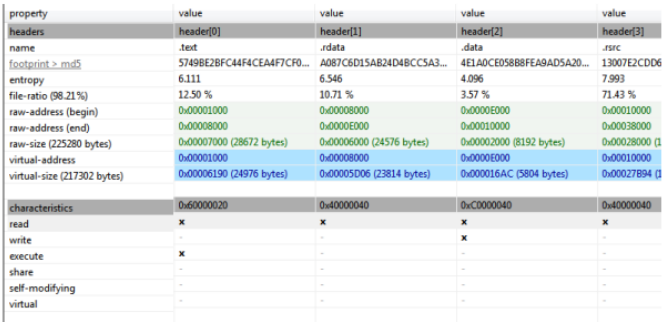


Fig. 3. Permissões dos diferentes módulos.

TABELA I

RELAÇÃO ENTRE BIBLIOTECAS E NÚMERO DE `IMPORTS` VERIFICADOS PARA A AMOSTRA DO *ransomware* WANNACRY.

| Biblioteca | `Imports` | Descrição |
|---|---|---|
| KERNEL32.dll | 39 | Windows NT BASE API Client |
| USER32.dll | 1 | Multi-User Windows USER API Client Lib |
| ADVAPI.dll | 5 | Advanced Windows 32 Base API |
| WS2 32.dll | 3 | Windows Socket Library |
| MSVCRT.dll | 39 | Microsoft C Runtime Library |

Além disso, é possível identificar os módulos responsáveis pela execução e pela criptografia dos arquivos, baseados nas *flags* de permissão `execute` e `write`, respectivamente, pois o componente de criptografia é quem realiza as alterações nos arquivos do sistema.

Através de um *plugin* dentro da ferramenta PeStudio, localizaram-se os `imports` que o executável realiza durante a sua rotina. Na Tabela I, apresentam-se as principais bibliotecas e números de `imports` verificados. Nota-se que o *ransomware* faz uso de DLL's nativas do Windows. Por exemplo, `KERNEL32.DLL` é uma DLL voltada a manipulação de me-

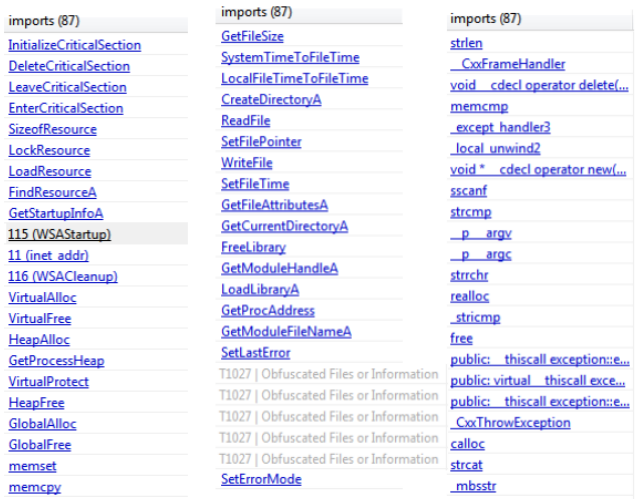


Fig. 4. Funções listadas pela ferramenta PeStudio na análise estática da amostra de *ransomware* WannaCry.

mória, arquivos e funcionamento do hardware. `ADVAPI.DLL` provê acesso a componentes avançados do windows como o registro e o gerenciador de tarefas. `USER32.DLL` contém todas as funções voltadas para o usuário do sistema operacional. O WannaCry faz o uso dessa biblioteca por se tratar de um executável e apresentar uma interface. `WS2 32.DLL` apresentam alguns recursos de rede, como conexão à Internet e outras funções. `MSVCRT.DLL` é outra biblioteca que o *malware* utiliza para manipulação de *strings* e manipulação de memória. Na análise não foi encontrada a presença de nenhum `import` de um possível componente "Worm". Logo, o módulo pode não estar presente na amostra ou foi ofuscado.

Na Figura 4, verificam-se as funções listadas pela ferramenta PeStudio. Em princípio, é possível observar várias manipulações de memória, registro e *strings* que derivam das bibliotecas apresentadas anteriormente. Além disso, há várias funções de exceção e tratamento de erros no WannaCry, bem como algumas funções ofuscadas. Diferente do empacotamento, a ofuscação não se encontra presente em grande parte das funções do *malware*, visto que os nomes das funções da amostra são legítimos e conhecidos da rotina de execução. Com o auxílio da ferramenta PEID, descobriram-se as principais funções do processo de criptografia do *ransomware*, conforme visto na Figura 5.

A função `CryptDecrypt` é utilizada para descriptografar os dados que são criptografados pelo *ransomware*. A função recebe como parâmetros a chave de criptografia, o *buffer* de dados criptografados e o respectivo tamanho. A função `CryptDestroyKey` destrói uma chave de criptografia da memória, dificultando a recuperação dos dados criptografados. Já a função `CryptReleaseContext` libera os recursos do sistema que estavam sendo utilizados pelo contexto de segurança, enquanto a `CryptImportKey` importa uma chave de criptografia para um contexto de segurança para que seja usada na criptografia e a armazena de forma segura. Por fim, a função `CryptAcquireContextA` obtém acesso ao serviço de criptografia desejado para viabilizar a criptografia.

O termo "contexto" se refere a um objeto que armazena

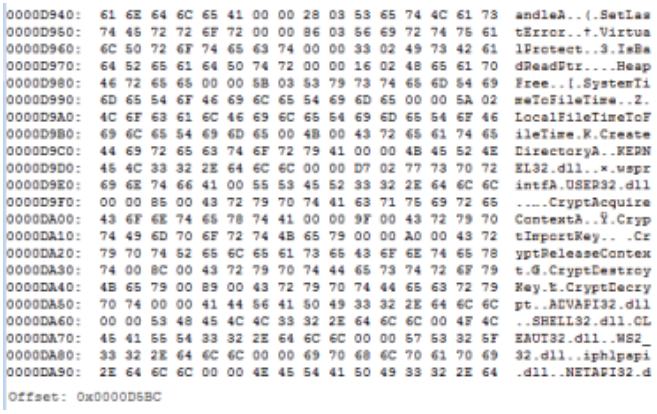


Fig. 5. Funções de criptografia verificadas para a amostra de *ransomware* WannaCry.


Fig. 6. Tela característica de resgate do *ransomware* WannaCry, vista segundos após a execução da amostra.

informações sobre o provedor de serviços de criptografia e as chaves de criptografia que estão sendo utilizadas [18]. Em outras palavras, o WannaCry obtém acesso ao serviço de criptografia desejado e armazena informações no contexto sobre o serviço de criptografia, como as referências das chaves de criptografia. Após esse procedimento, a chave é importada para o contexto de segurança, a criptografia é executada, as chaves são destruídas da memória e os recursos de sistema utilizados no contexto de segurança são liberados.

## IV. Resultados da Análise Dinâmica

Durante a análise dinâmica, o ambiente foi preparado e as ferramentas Procmon e Wireshark foram configuradas em modo *listening* para coleta de evidências das interações que foram levantadas durante a análise estática. Na sequência, executou-se a amostra para observar seu comportamento. Visualmente, verificou-se o processo de criptografia dos arquivos e alteração de suas extensões, bem como a criação de alguns novos arquivos na área de trabalho e a exibição sequencial de uma tela de resgate, conforme visto na Figura 6.

Inicialmente, avaliaram-se os novos arquivos criados e sua relação com algum módulo da amostra. O arquivo `!WannaDecryptor.exe` mediante execução, serve

para mostra a tela de resgate. Os três arquivos gerados `00000000.pky`, `00000000.res` e `00000000.eky`, se submetidos ao mesmo procedimento da análise estática, correspondem às chaves de criptografia e descriptografia da seguinte forma: `00000000.res` provê detalhes a respeito do servidor externo que os atacantes usam para comunicação (ou exfiltração de dados) através do *malware* (*Command Control server*, C2), com um cabeçalho apresentando informações genéricas da criptografia e sem conteúdo; e os arquivos `00000000.eky` e `00000000.pky` são chaves utilizadas pelo próprio Wannacry, sendo respectivamente a chave privada RSA e a chave pública RSA.

O processo de criptografia implementado pelo Wannacry utiliza dois algoritmos de criptografia. A parte principal do processo de criptografia utiliza um algoritmo de criptografia simétrica AES para criptografar todos os arquivos usando a função `CryptGenRandom`. Posteriormente, a criptografia assimétrica usa o algoritmo RSA para criptografar a chave de criptografia simétrica. Caso a vítima efetue o pagamento do resgate, ela recebe uma chave de decriptografia para que o WannaCry execute uma rotina que é capaz de verificar a presença dos arquivos mencionados e proceder com a decriptografia. Nesse caso, o Wannacry usa sua chave privada RSA para descriptografar a chave simétrica e, sequencialmente, utiliza-a para decriptografar os arquivos. Os arquivos `c.wry`, `u.wry`, `t.wry`, `r.wry` e `m.wry` fornecem instruções adicionais para o processo de criptografia e decriptografia.

A análise realizada pela ferramenta Wireshark apontou que nenhuma tentativa de lateralização foi encontrada, confirmando a ausência do módulo de rede ou *worm* nessa amostra, conforme constatado durante a análise estática. Em relação ao sistema de arquivos, foram gerados milhares de alterações durante a execução do *ransomware*, com destaque para a modificação de arquivos (criação e escrita/criptografia), e o uso das DLL's e funções.

Durante a análise dinâmica, avaliou-se a capacidade da amostra de distinguir a extensão do arquivo no processo de criptografia. Por exemplo, alguns *ransomwares* evitam criptografar arquivos do sistema e chaves de registro para evitar danos à máquina e garantir seu funcionamento com intuito de incentivar o resgate financeiro. No geral, estes *malwares* fazem com que apenas documentos específicos sejam perdidos. Para a amostra WannaCry, observou-se que nenhuma distinção foi feita durante a criptografia dos dados, o que pode ocasionar inclusive danos na integridade do sistema da máquina.

## V. Conclusão

O estudo da amostra do *ransomware* WannaCry realizado por meio desse trabalho permitiu avaliar e compreender o seu funcionamento. Identificaram-se características cruciais tanto na análise estática quanto na dinâmica, permitindo uma compreensão mais ampla do *malware* e suas atividades, apesar da ausência de evidências claras de propagação em algumas instâncias. Durante a análise estática, foi possível identificar as dependências, funções e bibliotecas utilizadas pelo *ransomware*. Enquanto na análise dinâmica, verificaram-se atividades relacionadas a criptografia e alterações de arquivos. Embora cada amostra apresenta particularidades, os métodos descritos podem ser replicados para entendimento geral do funcionamento de outras amostras de *ransomware*, sejam variantes do WannaCry ou outra família. Trabalhos futuros visam a avaliação de outras amostras de *ransomware* para comparação e identificação de possíveis módulos de laterização. Essa característica viabiliza o estudo do comportamento do *ransomware* em ambientes com soluções em programas de *firewall* e sistemas de detecção e prevenção.

## Referências


[1] R. Richardson and M. M. North, "Ransomware: Evolution, mitigation and prevention," *International Management Review*, vol. 13, no. 1, p. 10, 2017.

[2] L. V. Casagrande, E. C. V. Boas, and G. P. Aquino, "Systems, software, and applications updating for avoiding cyber attacks: A pentest demonstration," *XL Simpósio Brasileiro de Telecomunicaçoes e Processamento de Sinais (SBrT2022)*, 2022.

[3] M. Akbanov, V. G. Vassilakis, and M. D. Logothetis, "Wannacry ransomware: Analysis of infection, persistence, recovery prevention and propagation mechanisms," *Journal of Telecommunications and Information Technology*, no. 1, pp. 113–124, 2019.

[4] A. Bhardwaj, V. Avasthi, H. Sastry, and G. Subrahmanyam, "Ransomware digital extortion: a rising new age threat," *Indian Journal of Science and Technology*, vol. 9, no. 14, pp. 1–5, 2016.

[5] S. Yu, G. Gu, A. Barnawi, S. Guo, and I. Stojmenovic, "Malware propagation in large-scale networks," *IEEE Transactions on Knowledge and data engineering*, vol. 27, no. 1, pp. 170–179, 2014.

[6] M. Anghel and A. Racautanu, "A note on different types of ransomware attacks," *Cryptology ePrint Archive*, 2019.

[7] M. S. Smith and et al., "Ransomware-as-a-service: A growing threat to businesses and individuals," *Cybersecurity*, vol. 6, no. 2, p. 11, 2023.

[8] N. K. Popli and A. Girdhar, "Behavioural analysis of recent ransomwares and prediction of future attacks by polymorphic and metamorphic ransomware," in *Computational Intelligence: Theories, Applications and Future Directions-Volume II: ICCI-2017*. Springer, 2019, pp. 65–80.

[9] J. R. S. Alrzini and D. Pennington, "A review of polymorphic malware detection techniques," *International Journal of Advanced Research in Engineering and Technology*, vol. 11, no. 12, pp. 1238–1247, 2020.

[10] Q. Chen and R. A. Bridges, "Automated behavioral analysis of malware: A case study of wannacry ransomware," in *2017 16th IEEE International Conference on machine learning and applications (ICMLA)*. IEEE, 2017, pp. 454–460.

[11] S. Mohurle and M. Patil, "A brief study of wannacry threat: Ransomware attack 2017," *International journal of advanced research in computer science*, vol. 8, no. 5, pp. 1938–1940, 2017.

[12] M. Sikorski and A. Honig, *Malware Analysis: A Hands-On Guide*. No Starch Press, 2012.

[13] E. H. Härder and M. Tuexen, "Static analysis of malware: A survey," *Journal of Computer Virology and Hacking Techniques*, vol. 9, no. 2, pp. 115–138, 2013.

[14] W. E. et al., "Dynamic analysis of malware: A survey," *IEEE Security & Privacy*, vol. 9, no. 5, pp. 58–66, 2011.

[15] H. Dwivedi and A. K. Sahu, "Detecting sandbox environments by malware: A survey," *Journal of Information Security and Applications*, vol. 67, p. 103076, 2023.

[16] D. Song, Y. Ye, W. Li, X. Wang, and W. Zhang, "Entropy as a measure of complexity and randomness in malware analysis," *International Journal of Security and Its Applications*, vol. 5, no. 4, pp. 249–258, 2011.

[17] (2021, Jun) How to unpack upx packed malware with a single breakpoint. [Online]. Available: https://infosecwriteups.com/how-to-unpack-upx-packed-malware-with-a-single-breakpoint-4d3a23e21332

[18] D. A. Mcgrew, J. Viega, and C. Boyd, "Cryptography: Engineering and design," 2014.